Guidelines for Implementing Computerized Logistics Management Information Systems (LMIS)

October 2003

DELIVER

DELIVER, a five-year worldwide technical assistance support contract, is funded by the Commodity Security and Logistics Division (CSL) of the Office of Population and Reproductive Health of the Bureau for Global Health (GH) Field Support and Research of the U.S. Agency for International Development (USAID).

Implemented by John Snow, Inc. (JSI), (contract no. HRN-C-00-00-00010-00), and subcontractors (Manoff Group, Program for Appropriate Technology in Health [PATH], Social Sectors Development Strategies, Inc., and Synaxis, Inc.), DELIVER strengthens the supply chains of health and family planning programs in developing countries to ensure the availability of critical health products for customers. DELIVER also provides technical support to USAID's central contraceptive procurement and management, and analysis of USAID's central commodity management information system (NEWVERN).

Recommended Citation

John Snow, Inc./DELIVER. 2003. *Guidelines for Implementing Computerized Logistics Management Information Systems (LMIS)*. Arlington, Va.: DELIVER/John Snow, Inc., for the U.S. Agency for International Development.

DELIVER
John Snow, Inc.
1616 North Fort Myer Drive, 11th Floor
Arlington, VA 22209 USA
Phone: 703-528-7474
Fax: 703-528-7480
Email: deliver_project@jsi.com
Internet: deliver.jsi.com

Contents

ACKNOWLEDGEMENTS **7**

INTRODUCTION **9**

What is a Computerized LMIS? 9

Guidelines Overview 10

RECOMMENDED COMPONENTS OF COMPUTERIZED LMIS **11**

Recommended Data 11

Recommended Functions 13

Recommended Reports 14

Recommended Graphs 16

DEVELOPMENT AND IMPLEMENTATION OF COMPUTERIZED LMIS **17**

Need for a Development and Implementation Process 17

Overview of a Process 18

Roles and Responsibilities of Process Participants 19

Analyzing Client Needs 21

Designing the LMIS 25

Designing the LMIS for Multiple Distribution Levels 28

Developing the LMIS 33

Buying the LMIS 36

Testing the LMIS 38

Documenting the LMIS 40

Training LMIS users (Performance Improvement) 42

Deploying the LMIS 43

Modifying the LMIS 44

COMPUTERIZED LMIS OPERATIONS **47**

Placing the Computerized LMIS Within the Central Organization 47

Managing Computerized LMIS Data 49

Providing Technical Support 50

GLOSSARY **51**

APPENDIX 1: LESSONS LEARNED IN IMPLEMENTING A COMPUTERIZED LMIS ... 53
Background 55
Organizational Context 57
Mechanisms and Resources for Software Maintenance and Modification 63
Utilization of LMIS Data for Decision Making **67**

APPENDIX 2: RECOMMENDED LMIS REPORTS AND GRAPHS 71
Stock Status by Distribution Level 73
Stock Status by Facility 74
Stock Status by Product 75
Stock Status Errors 76
Data Entry Errors 77
Non-reporting Facilities 78
Stock Imbalances 79
Adjustments Summary 80
Dispensed to Users 81
Stock Status 82
Average Stock Levels 8**3**

APPENDIX 3: LMIS DEVELOPMENT PROCESS DOCUMENTS 85
Requirements Assessment Outline 87
Requirements Assessment Example, Warehouse Management System (WMS) 89
Use Case 108
User Interface Prototype 109
Site Map 110
Database Entity-Relationship Diagram 111
Enhancement Specification 112
Test Case 113
User's Guide 115
Technical Manual 116
Training Curriculum 117

TABLES

1: Recommended LMIS Data 12
2: Recommended LMIS Functions 13
3: Recommended LMIS Reports 15
4: Recommended LMIS Graphs 16
5: Computerized LMIS Development Roles and Responsibilities 20
6: Use Case Extensions and Examples 22
7: User Interface Prototypes 27
8: LMIS Database Configuration Options 32
9: Software Objects 34
10: Two-pronged Training Method for a Computerized LMIS 43
11: Data Items and Reports in a Defect and Enhancement Tracking Tool 46
12: Types of Severity of Software Defects 46
13: Advantages and Disadvantages of Location of the Computerized LMIS 48

FIGURES

1: Central Database 30
2: Distributed Databases 31
3: Central Online Database 32

BOX

1: Use Case Example 24

Acknowledgements

These guidelines benefited from contributions by Kip Eckroad, Ashraf Islam, Karen Ampeh, Larry Bailey, and Edward Wilson. The following people also reviewed and contributed ideas to the section on computerized LMIS development and implementation: Lisa Blankenship, Jeff Leiner, Gideon Nzoka, John Zingeni, Mercy Maina, Moses Muwonge, Kim Peacock, and Laurie Lyons. Barbara Felling, Bill Felling and Edward Wilson reviewed and commented on draft versions of the guidelines.

Leslie Rock is the primary author of the guidelines.

INTRODUCTION

WHAT IS A COMPUTERIZED LMIS?

A logistics management information system (LMIS) collects, processes, and reports logistics data. A well-functioning LMIS provides decision makers throughout a supply chain with accurate, timely, and appropriate data. The LMIS can be manual (paper-based), partly computerized, or entirely computerized.

In a computerized LMIS, computers take the place of humans in aggregating logistics data—performing calculations and producing reports and graphs for analysis. A computerized LMIS provides several important benefits over a manual LMIS:

- **No mathematical errors.** Unlike their human counterparts, computers never make mathematical errors. Human LMIS operators may make calculation errors that cause them to report inaccurate movement of stock, stock on hand, and rates of consumption, which may lead to forecasts and procurement plans that are wildly off base. A well-designed computerized LMIS includes methods for the computer to validate manual calculations submitted on LMIS reports.
- **Rapid aggregations and calculations.** Not only do computers calculate and aggregate logistics data with 100 percent accuracy, but they also perform the calculations and aggregations rapidly. Rapid processing means that logistics information is available to managers and decision makers in a more timely manner.
- **Rapid production of reports and graphs.** Computers can also produce reports and graphs on the spot to meet the information needs of logistics managers and decision makers. Graphs, in particular, provide valuable, at-a-glance information to high-level managers who do not have time to peruse reams of LMIS reports.

At the same time, there are disadvantages associated with computerized LMIS:

- **Dependence on design.** Even more than the logistics systems it serves, a computerized LMIS relies on a good design. A design that is rushed or poorly thought out can ruin a computerized LMIS. A poor design may not be immediately visible to logistics managers, and it may make the LMIS awkward to use or, worse, cause the LMIS to provide wrong information.
- **Dependence on technology.** Computerized LMISs needs more than a steady supply of pens and paper to stay in operation; they need computer hardware, printers, data backup mechanisms, and reliable electricity. All these things cost money, which may reduce the feasibility of a computerized LMIS in many public health organizations.
- **Dependence on technicians.** To keep the software in good working order, successful computerized LMIS rely not only on hardware but also on regular support from computer technicians. Logistics managers generally cannot learn how to provide this support by participating in a short course. Without locally available technical support, a computerized LMIS is likely to become unusable.

It is important to consider the advantages as well as the disadvantages when deciding whether to introduce a computerized LMIS.

GUIDELINES OVERVIEW

These guidelines cover several key topics related to a computerized LMIS. The next chapter lists recommended components of a computerized LMIS: LMIS data, functions, reports, and graphs. Together, these components represent the basic elements of a well-designed computerized LMIS.

The third chapter outlines a process for developing and implementing a computerized LMIS. The process encompasses a number of essential activities, including analysis of information needs, LMIS design, software testing, and user training. Without a defined development and implementation process, a computerized LMIS project runs the risk of failure.

The fourth chapter describes the activities required for computerized LMIS operations after the LMIS has been implemented. LMIS operations include routine data management as well as technical support. The section also addresses the placement of a computerized LMIS in an organization's central office, and weighs the advantages and disadvantages of several options.

A computerized LMIS at multiple levels of a distribution system is also addressed in chapter four. To be most effective, a computerized LMIS at multiple distribution levels requires an electronic exchange of LMIS data among levels. Several options for configuring a computerized LMIS to enable electronic data exchange are discussed and evaluated.

Appendix 1 presents lessons learned from computerized LMIS developed and implemented by John Snow, Inc.'s (JSI) FPLM project in five countries: Bangladesh, Jordan, Kenya, Nepal, and the Philippines. The lessons are organized into three areas: organizational context, mechanisms and resources for software maintenance and modification, and utilization of LMIS data for decision making.

Appendix 2 displays template reports and graphs for a computerized LMIS.

Appendix 3 includes a number of template documents for use during the process of developing and implementing a computerized LMIS. Readers are free to use these documents as models.

Recommended Components of a Computerized LMIS

RECOMMENDED DATA

For any logistics system, the three essential LMIS data items are (1) quantity of stock on hand, (2) quantity of stock consumed (dispensed to users), and (3) losses and adjustments. In addition, a computerized LMIS needs to track data on the facilities that receive and distribute products and on the products that are distributed. The LMIS also needs to gather data via logistics reports collected on a scheduled basis—quantities of each product received, issued, and dispensed to users.

Recommended data groups and items are listed in table 1.

CALCULATED DATA ITEMS

Some data items can be calculated: stock on hand, available months of stock, and average monthly consumption are the most commonly calculated items. The results of these calculations can be stored in the database of a computerized LMIS, but it is generally preferable not to store calculated data in a database. Calculations depend on other data items and are, therefore, subject to change whenever the values of the other data items change. It is best to perform the calculations as needed for display on a computer screen or in reports.

One major exception is the requirement to validate logistics data reported by facilities. If the software needs to check that logistics managers at a facility are performing the calculations correctly, it may be desirable to store the reported value and compare it with a computer-generated calculation.

Table 1: Recommended LMIS Data

Data Group	Data Items

Logistics system	▪ Reporting/delivery period (monthly, quarterly or other)
	▪ Number of periods to use in calculating average monthly consumption
	▪ Type of calculations performed—
	  ▫ Computer-generated
	  ▫ User-generated
	  ▫ User-generated with computer validation
	▪ Unit of measure (English or metric)

Facilities	▪ Facility name ▪ Facility address ▪ Contact person ▪ Facility type ▪ Supplying facility ▪ Distribution role (warehouse or SDP) ▪ Distribution level ▪ Active/inactive
Facility types	▪ Facility type name ▪ Maximum and minimum months of stock
Products	▪ Product name ▪ Product category ▪ Dispensing unit ▪ Active/inactive ▪ Indicator/not indicator
Logistics reports	▪ Report name ▪ Reporting period ▪ Facility reporting ▪ Opening balance (optional) ▪ Stock on hand ▪ Issues/quantity dispensed to users ▪ Receipts ▪ Losses and adjustments

RECOMMENDED FUNCTIONS

Recommended functions that a computerized LMIS should carry out are listed in table 2.

Table 2: Recommended LMIS Functions

Function	Description
Manage facilities	Enable users to add, edit, and inactivate facilities in the logistics system.
Manage products	Enable users to add, edit and inactivate products in the logistics system.
Manage logistics data reported by each facility on a scheduled basis	Enable users to add, edit and delete individual logistics reports submitted by facilities.
Aggregate logistics data for each product	For each product, sum the receipts, issues, consumption and stock on hand reported by facilities over a period of time specified by the user.
Aggregate logistics data for each facility	For each facility, sum the receipts, issues, consumption and stock on hand by product over a period of time specified by the user.
Calculate consumption averages	For each facility and product, calculate average monthly consumption based on several consumption reports over a period of time.
Calculate months of stock	For each facility and product, divide the reported (or calculated) stock on hand by the calculated average monthly consumption.
Calculate quantities to resupply	Multiply the calculated average monthly consumption by the maximum stock level and subtract the reported stock on hand.
Calculate reporting rates	For each reporting period, calculate the percentage of facilities that have submitted reports.
Identify non-reporting facilities	For each reporting period, identify facilities that have not submitted reports.
Identify potential data errors	Verify that— ▪ Facilities do not report receiving more or less in aggregate than what their supplying facility reported issuing. ▪ A facility's reported opening balance matches its ending balance from the previous reporting period.

RECOMMENDED REPORTS

Reports are the main outputs of a computerized LMIS. Logistics managers use them in databased decision making, and high-level managers may rely on them to implement policies affecting the national supply chain. But, the process of implementing a computerized LMIS often focuses on data entry operations—in other words, how to get data into the computer rather than how to get data out of it. An approach to LMIS implementation that focuses on reports as well as data entry stands a better chance of meeting the needs of higher-level decision makers. (A two-pronged approach to training both data entry operators and users of LMIS data is discussed on page 43 the Training LMIS Users [Performance Improvement] section.)

Reports are designed to answer the following key questions about the stock status of a particular facility or product, and about the overall performance of the logistics system.

- What is the national stock status for a particular product?
- What is the stock status of a particular facility?
- How much of a particular product should be resupplied to a facility?
- What is the status of reporting for a particular period?
- Are there any errors in reporting by facilities?
- Are there any errors in data entry?
- Are any facilities not stocked according to plan—i.e., overstocked, understocked, or stocked out?
- Are there any unusual patterns of losses or adjustments for a particular product or facility?

Recommended LMIS reports are listed in table 3. Samples of recommended LMIS reports are included in appendix 2.

Table 3: Recommended LMIS Reports

Report	Description
Stock status by distribution level	*Summarizes the stock status for one or more selected products at each distribution level in the logistics system*. This report answers the questions, "What is the national stock status for a particular product?" and "What is the stock status at a particular distribution level for a particular product?"
Stock status by facility	*Lists the stock status of all products for each facility in the logistics system*. Logistics managers may use this report to determine whether a facility is stocked according to plan.
Stock status by product	*Shows the status of a particular product at all facilities in the logistics system*. Logistics personnel may give this report to program managers to monitor the status of one or more products used by their program.
Supply status errors	*Identifies potential errors in reporting by facilities*. When the reported issues from a supplying facility don't match the sum of receipts for the facilities it supplies, the report flags those records for investigation.
Data entry errors	*Notes potential data entry errors*. It flags facility reports in which one or more columns contain a math error that caused the computer to generate an adjustment, allowing the software operator to first validate his/her data entry before contacting the facility to resolve the error.
Non-reporting facilities	*Identifies facilities that have not submitted reports for a selected reporting period*. Software operators or managers may view this report frequently during the reporting period to contact facilities whose reports have not arrived. This report may also be useful in forecasting future needs based on logistics data; forecast quantities can be adjusted in cases where data are from fewer than 100% of facilities.
Stock imbalances	*Indicates imbalances in the supply status at every facility in the distribution system: overstocks, understocks, and stockouts*. Managers can use this report to identify facilities that are not stocked according to plan and take action to correct the imbalances.
Adjustments summary	*Summarizes adjustments by product and by adjustment type*. Logistics managers may view this report to identify any unusual adjustments for a particular product or by a particular facility.

RECOMMENDED GRAPHS

Graphs can convey essential information on logistics system performance in a quickly understood visual format. This format is particularly suited to high-level decision makers who do not have time to analyze tabular reports.

Table 4 lists recommended graphs for a computerized LMIS. Sample graphs are also included in appendix 2.

Table 4: Recommended LMIS Graphs

Graph	Description
Dispensed to Users	*Displays quantities of a selected product dispensed to users over a selected period of time.* This graph can be displayed as either a line or bar graph showing quantities dispensed over time.
Stock Status	*For a selected product, displays percentage of facilities that are stocked according to plan over a selected period of time.* May also include percentage of facilities stocked out or overstocked. This graph is best displayed as a stacked bar graph in which each type of status (within plan, stocked out, and overstocked) is a component of the bar. It can be further refined by setting the height of each bar to the reporting rate percentage during that period.
Average Stock Levels	*Displays the average stock level of a selected product over a selected period of time.* This graph can be displayed as a line or bar graph showing average stock level expressed as number of months of stock.

Development and Implementation of a Computerized LMIS

Need for a Development and Implementation Process

A defined development and implementation process is essential to the success of any computerized LMIS. It is essential for three primary reasons: (1) communication, (2) quality assurance, and (3) learning.

- *Communication.* Computerized LMIS implementation requires the active participation of many people—sponsors, end users, developers, testers, and product managers. A clearly understood process facilitates communication among these participants, and is especially important when participants are geographically dispersed.
- *Quality assurance.* Implementing a computerized LMIS is a complex task and prone to errors. The later these errors are discovered, the more costly they are to fix. A process that includes reviews of outputs at every stage of the project can reduce costs and improve software quality by detecting and correcting errors early on.
- *Learning.* The most difficult aspect of computerized LMIS development is determining what the software should do. Typically, this learning process spans the entire life of the project. A well-documented process allows project participants to incrementally adapt the software development and implementation to best meet users' needs and objectives.

OVERVIEW OF A PROCESS

The widely recognized need for software implementation processes has spawned hundreds of prepackaged methodologies during the past 30 years. These methodologies typically attempt to prescribe in detail every step in a defined process. While they vary greatly in their approaches and techniques, methodologies generally include the following basic activities:

- gathering client needs
- analyzing client needs
- designing a software solution
- building the solution (or finding a packaged solution)
- testing the software
- writing reference materials
- deploying the software
- training users.

This process starts as soon as the need for a new or improved information system has been identified. The existing information system may be manual, computerized, or a combination of both. And the need for a new or improved system may be identified by the client, an outside consultant, or a technical assistance provider who later participates in development or implementation of the software. Regardless of the type of existing information system and the impetus for considering a new or improved system, the client must be involved from the start and continue throughout the process.

This chapter describes a software implementation process that encompasses the activities listed above and that can be used to implement a computerized LMIS. The section includes justifications for completing each activity and a brief description of the activity. Appendix 3 includes sample documents that support each activity.

ROLES AND RESPONSIBILITIES OF PROCESS PARTICIPANTS

The process of developing and implementing a computerized LMIS depends on the contributions of a number of participants with important roles to play:

- *Project sponsors* provide funding and may also provide high-level managerial support to the computerized LMIS project.
- The *product manager* facilitates and coordinates all software development and implementation activities.
- *Clients* are the intended users of the software or its outputs. They work with the other participants listed here to define or redesign business processes and to specify what the software must do to support those processes.
- *Developers* analyze information needs in collaboration with the other participants, then design and build software to meet those needs.
- *Testers* verify that the software does what it is designed to do, and tells the developers when it doesn't work correctly.
- *Documenters* develop reference materials for end users and technical support personnel. (The technical support function is described in the next chapter.)

In many cases, participants play more than one role; for example, clients often serve as testers of the delivered software before the software is implemented. In some cases, one role is shared among several people: in larger projects, the developer role may be played by a team of people—led by the product manager—who analyze needs, design a software solution, and then build the solution. Regardless of the number of people involved in the project, it is essential that all the roles are filled; otherwise, the software project is likely to fail.

Table 5 displays the roles and responsibilities of project participants.

Table 5: Computerized LMIS Development Roles and Responsibilities

Role	Likely Participants	Responsibilities
Project Sponsor	▪ High-level managers within the client organization ▪ International donors or donor representatives	▪ Provide funding and other resources required by the project. ▪ Review and provide feedback on key outputs during the course of the project.
Product Manager	▪ Manager within the client organization ▪ Donor representative	▪ Allocate project resources. ▪ Assign participants to project activities. ▪ Monitor and review project activities. ▪ Facilitate communication among project participants.
Clients	Within the client organization— ▪ Data entry personnel ▪ Logistics personnel ▪ Health program personnel	▪ Share detailed information on current procedures and information needs. ▪ Review and provide feedback on outputs during the course of the project.
Developers	▪ Individual consultant ▪ MIS unit personnel ▪ IT company personnel	▪ Collaborate with product manager and end users to identify and prioritize software requirements. ▪ Design and build software according to identified requirements. ▪ Communicate with product manager and end users about feasibility, cost, and resource needs of requirements.
Testers	▪ Data entry personnel ▪ MIS unit personnel ▪ IT company personnel ▪ Program personnel	▪ Test software to confirm that it meets requirements. ▪ Find bugs or problems and report them to developers. ▪ Verify bug fixes.
Documenters	▪ Product manager ▪ IT company personnel	▪ Develop user's manuals or job aids for end users. ▪ Develop technical documentation for technical support personnel.

ANALYZING CLIENT NEEDS

Analysis is the first activity in the process of implementing a computerized LMIS. The major goal of analysis is to answer in detail the following questions:

- How does the existing LMIS work?
  - How is the information collected, processed, and presented?
  - What decisions are being made and at what level?
  - What information is needed to make those decisions?
- Who uses the LMIS and why?
- What problems exist with the LMIS?
  - Why do these problems exist?
  - What are their root causes?
- What is needed to solve the problems with the existing LMIS?
  - Is a new or improved LMIS the best solution?
  - Is the best solution manual or computerized?

Analysis is often the most challenging activity in computerized LMIS implementation, because the root problems of an existing LMIS are difficult to uncover and because requirements for a computerized LMIS are equally elusive. When the client and sponsor initiate a software project, they have a sense that something is wrong or needs improvement, but find it difficult to articulate those problems and to specify what the new system should do. During analysis, the task of the software team is to elicit these requirements in collaboration with the client.

You may find reviews of the existing information system in previous assessments of the logistics system. If no previous studies have been conducted, the *Logistics Management Information Systems Assessment Guidelines* suggest a process for reviewing the current LMIS design and operations.

If a computerized LMIS is the recommended solution, then the first sample process document—the requirements assessment—specifies sections for the written output of an analysis or requirements gathering exercise. This document is shared with the client and sponsor as a way to clarify users' business objectives and to gain agreement on the focus and scope of the proposed software implementation. Sharing the requirements assessment with the client at this early stage enables the client to provide feedback to the team so errors or misunderstandings are eliminated well before they are written into software code.

Two of the most valuable sections of the requirements assessment are the system functions and the illustrative outputs. The illustrative outputs present sample reports using real data where available. These sample reports are useful in letting the users see what they will get out of the system. The system functions, or use cases, depict the interaction between the users and the new, computerized system to accomplish a particular objective. It is not necessary to cover every single possible use case in the requirements assessment—most software applications ultimately encompass hundreds of use cases—but only those that represent the basic functions of the computerized system, such as receiving and issuing products. Guidelines for writing uses cases are included in this section.

Use Cases

A use case shows a user and a computerized system interacting to reach a goal. Use cases are meant to be read by a potentially wide range of team members and are therefore written in text prose, unaccompanied by technical symbols or notations. The first step in writing a use case is naming the case, using a simple, action-oriented, verb-object phrase. Examples of use cases for LMIS or warehouse management systems (WMS) include *receive products* or *issue products*. A sample use case is displayed in box 1 on page 24.

Use Case Extensions

The *main course of events* section of the use case is often alternatively called the *success scenario*. Because many interactions do not end in success but rather failure, or end in another way, how do we document these other interactions? A separate section of the use case—*extensions*—provides a place to address how the user and system will handle unexpected or alternative events. Table 6 lists possible ways the main scenario can fail, followed by examples.

Table 6: Use Case Extensions and Examples

Use Case Extension	Example
Alternate success path	User types a shortcut code
Incorrect user behavior	Invalid password
User inaction	Time-out waiting for password
Any time the phrase *the system validates* appears in the main scenario, implying that there will be an alternate path if the validation fails	Invalid item number
Unusual or failed response from a supporting system	Time-out waiting for response
Internal failure within the system under design that must be detected and handled as part of normal business	Incomplete or missing data in the database
Unexpected and abnormal internal failure that must be handled	Corrupt database
Critical performance failures of the system that must be detected	Response not calculated within ten seconds

Guidelines for Writing Use Cases

Above all, use cases are written in a clear and concise manner to represent users' goals and computerized system actions to reach these goals. The following guidelines list rules-of-thumb for focusing the use cases solely on user and system actions and, at the same time, eliminating wordiness. [1]

- *Use simple grammar.* Short, action-oriented, subject-verb-object sentences are the clearest way to present the story.
- *Show clearly "who has the ball."* Include a subject in all your sentences. Otherwise, others may have a hard time deciphering who does what to whom.
- *Show the process moving forward.* Don't include minute actions (e.g., "user hits tab key") that slow down the use case.

[1] Cockburn, Alistair. 2001. *Writing Effective Use Cases.* Boston: Addison-Wesley.

- *Show the user's intent, not the movements.* Don't describe the user's movements in operating the user interface; instead, focus on the user's intent in interacting with the system. This strips out extraneous—and premature—user interface specifications.
- *Include a reasonable set of actions.* Each step in a use case represents a transaction, which contains four parts. Look for opportunities to combine these parts into less than four steps, which makes the use case easier to read and saves space.
- *"Validate," don't "check whether."* The main use case is a success scenario; don't include tasks that lead to use case failure. Instead, include failure points in the extensions section.
- *Idiom: "User has System A alert System B."* Many system requirements include the need to interact with other computerized systems. This phrase is the simplest way to convey that the user has control over the interaction, without specifying how the interaction is initiated.
- *Idiom: "Do steps X-Y until condition."* In many use cases, one or more steps can be repeated. List this repetition at the end of the step or steps being repeated; to make the use case easier to read, do not number the repetition.

Appendix 3 includes an outline for a requirements assessment, a sample assessment, and a sample use case.

Box 1: Use Case Example

Manage Logistics Data Reported by a Facility

Summary

On a scheduled basis (monthly, quarterly, or other time period), facilities in the distribution system submit logistics data. As these data are received, the LMIS operator records them in the computerized LMIS.

Main course of events

1. Operator selects the facility that submitted the report.
2. Operator selects the reporting period listed on the report (specific month, quarter, or other time period).
3. System finds and displays all products authorized for distribution by that facility, and displays an area for data entry.
4. Operator selects a product listed on the report.
5. In the data entry area, operator enters the following for the product, as listed on the report:
   - opening balance
   - quantity received
   - quantity dispensed.
6. [Optional] Operator enters quantity adjusted, and selects an adjustment type.
7. Operator enters the closing balance as listed on the report.
8. System validates that all entered quantities balance.
9. System calculates and displays average monthly consumption as an integer, as follows:

$$\frac{\text{sum of [quantity dispensed] (over a specified number of periods)}}{\text{[number of periods]}}$$

10. System calculates and displays months of stock as a number with one decimal point (for example, 1.3) as follows:

$$\frac{\text{[closing balance]}}{\text{[average monthly consumption]}}$$

11. System calculates and displays quantity to resupply as an integer, as follows: ([average monthy consumption] x [maximum stock level]) - [ending balance]
12. Operator enters data for each additional product listed on the report.

Extensions

8a. Reported opening balance does not match ending balance from previous report:
   .1 System alerts operator that opening balance does not match, and prompts operator to re-enter opening balance.

8b. Reported quantities dispensed greater than stock available (opening balance + quantity received +/- quantity adjusted):
   .1 System alerts operator that quantity dispensed is greater than stock avail able, and prompts operator to re-enter quantity dispensed.

8c. Reported closing balance does not balance with other entered quantities:
   .1 System alerts operator that closing balance does not balance with other entered quantities, and prompts operator to re-enter closing balance.

DESIGNING THE LMIS

After the project participants agree on the requirements for the new, computerized system, developers begin to create more detailed descriptions for how the new system will look and act—descriptions or template images of all user screens, reports, and other user interface objects, as well as the database underpinning it all. (Often, developers begin this work before the requirements have been finalized, but they should always update their work based on any subsequent changes to the requirements.) These descriptions are documented in the following documents:

- *Database entity-relationship diagram.* This document shows the structure of the database: major data groups (entities), relationships between these groups, and the data items (attributes) composing each group. A sample entity-relationship database diagram is displayed on page 109.
- *User interface prototype.* The user interface prototype is a snapshot of one or more screens that the user will see when operating the computerized LMIS. User interface prototypes are discussed later in this section. A sample user interface prototype is displayed on page 107.
- *Site map.* The site map shows the relationship among screens in the user computerized LMIS. The site map is useful in communicating to users where particular LMIS functions will be accessed. A sample site map is displayed on page 108.
- *Sample reports.* Sample reports depict the expected outputs of the computerized LMIS. High-level logistics managers and decision makers may never see the software itself but will only see its outputs. Showing these key users sample reports during the design phase can bolster high-level support for the development process. Appendix 2 includes sample LMIS reports.
- *Business rules.* Business rules are the critical behind-the-scenes rules that the computerized LMIS must follow; for example, adherence to the product list managed by the central medical stores. In many cases, business rules can be incorporated into the use cases. Some business rules may need to be documented separately if they don't directly relate to the user interactions described in the use cases.

Of these, only the user interface prototype, site map, and sample reports might be shared with end users and project sponsors. Together, these documents act as a two-dimensional prototype of the new system, enabling users to see how the system will look and what it will produce. At this point, users can provide invaluable input to developers to modify the design, well before the prototype becomes a full-blown application and changes could require costly modifications.

Designing the Database

If project participants decide to develop customized software, the next major focus of their work is designing the database that stores the users' data. Several participants, who can find and fix potential design flaws, should carefully review any proposed design for the database. Software based on a flawed database may end up containing numerous data errors that cause the system to crash or, worse, perform erratically without the user's knowledge.

Databases should be designed to meet *third normal form*, which applies to every table in the database:

- All columns in each table must be atomic or contain only one value (first normal form).
- Every non-key column in each table must be fully dependent on the (entire) primary key (second normal form).
- All non-key columns in each table are mutually independent, i.e., no columns contain calculations or values that are dependent on the contents of other columns (third normal form).

Selecting the Programming Language

When choosing a programming language, project participants tend to go with the language they know best. The main advantage to this approach is that the developers' learning curve is much shorter or even nonexistent. But, the best-known language may not be the best option in all software development projects. Before selecting a programming language, project participants should attempt to answer the following questions—

- What is the client's standard programming language (if any)?
- What language meets most, if not all, of the requirements of the software to be developed?
- Is knowledgeable, local technical support available, other than the original developer? This support can be information technology staff within the client organization, individual consultants, or software firms based locally.

While the answers to all three questions are important, the answers to the first and third bullet should be weighted most heavily, because they indicate the capability of the client and the local environment to adopt and support the software.

Defining Minimum Requirements

Any software application—including computerized LMIS—has some minimum set of hardware and software requirements. The two most basic requirements, listed below, should be defined during the design phase, so developers know the required parameters.

- *Operating system.* Microsoft Windows 95®, 2000, or NT are the most common operating systems to design for.
- *Screen resolution.* In many resource-constrained environments, the highest available screen resolution may be 800 x 600. Developers should plan to design user interfaces that are readable at the highest screen resolution that will be available.

Creating a Prototype

Developing a prototype of the user interface is the major activity in the design phase. A prototype depicts interface elements—computer screens, paper reports, and other inputs or outputs—to show the users how the application will look and act. Meeting with users to review one or more prototypes is often one of the most productive activities in the process: users can react to what they see and can suggest in concrete terms how to improve it,

and developers can then modify the design based on those suggestions.

While prototypes are a productive way to further clarify users' requirements, they may also present the illusion of finished software to both users and developers. High-fidelity and/or vertical prototypes may simulate the proposed software so realistically that users and sponsors are fooled into thinking that all the development work is done. Developers may be pressured to continue to work on the prototype in order to deliver it as the operational system to users as soon as possible. This is a dangerous approach to development. Prototypes are patched together quickly to show users a pretty facade, with little or no attention given to critical issues behind the scenes. In most cases, prototypes are an unsuitable basis for operational systems.

What Prototype Is Best for Communicating with Users?

The main purpose of a prototype is to present a possible user interface to users before developers start to write code. Prototypes are developed quickly and then thrown away (or kept as documentation) before the programming starts, so they should be developed as cheaply as possible. For almost all software projects, a passive, paper-based prototype suffices to give users a picture of the proposed user interface, and can also be developed using standard word processing or graphics software or even pen and paper.

Making a passive prototype low-fidelity and horizontal can speed development of the prototype while keeping costs down. But, sometimes developing a high-fidelity and/or vertical prototype can greatly benefit the programming work that is about to begin. High-fidelity prototypes may be effective in demonstrating the proposed system to high-level managers or project sponsors to secure resources needed for development. And, vertical prototypes may be developed to fully explore and resolve particular software functions that remain unclear.

Table 7: User Interface Prototypes

Prototype Dimension	Variants
Interaction	▪ *Passive prototypes* are two-dimensional drawings of the user interface. They depict the interface but do not allow for any interaction with the user. ▪ *Active prototypes* allow for some user interaction, and are, therefore, also called functional or working prototypes or simulations.
Fidelity	▪ *High-fidelity prototypes* closely resemble the user interface, down to the smallest detail. ▪ *Low-fidelity prototypes* only vaguely look like the application they are designed to model. They may significantly assist communication with users in developing applications whose requirements remain vague.
Scope	▪ *Horizontal prototypes* show a broad but shallow horizontal slice of the application, and they are usually also passive prototypes. ▪ *Vertical prototypes* show by contrast a narrow but detailed vertical slice through the application.

DESIGNING THE LMIS FOR MULTIPLE DISTRIBUTION LEVELS

Electronic Exchange of LMIS Reports

Many software applications involve communication of some sort: communicating with the database over a network, exchanging data among different installations of the software, or sharing data with other applications, such as an accounting system or health management information system (HMIS). Developing and testing these communication links is surprisingly complicated and time-consuming. A general rule of thumb is to *communicate electronically only when necessary*. If there is a non-electronic way to convey the same information—via a paper report, for example—then project participants should choose that option first.

There are cases when electronic communication offers significant benefits over other forms of communication. JSI is currently considering developing a two-tiered version of a computerized LMIS in which district medical stores would exchange data electronically with the central medical store. The alternate, paper-based method would require each district to fill out and send to the central medical store paper copies of reports for the hundreds of health products consumed by every health center it supplies. This will likely cause a potentially lethal bottleneck at the central medical stores, which has to process reports from each and every health center throughout the country. In this case, electronic exchange of data between the region and center would help avoid the bottleneck, and is thus worth the extra time and cost to develop up front.

Questions to consider when developing communication links—

- *Who manages what data?* Exchanging data between two different installation sites or two different software applications can result in bad data if all sites can add, update, or delete the same data. If two or more sites change the same data record, whose record is the final record? Before setting up a two-way exchange, the project team must clearly define which installations have the right to add, update, or delete data. The assigned rights must be mutually exclusive, so that any changes to data are sent in one direction only.
- *When are data transmitted–on a regular or ad hoc basis?* In an LMIS, reports are sent on a regular basis—usually monthly or quarterly. Typically, some facilities report later than the cutoff date for submitting aggregated reports up the supply chain. How does the software transmit reports that have arrived behind schedule? Do those reports wait until the next scheduled reporting period, or are they sent as soon as they are received?
- *What data are transmitted–all data or only data that have changed since the last transmission?* In many software applications, users not only add new data but also modify or delete existing data. How are all these changes transmitted? Sending the entire database is the safest way to ensure that all data changes are transmitted. But, if data are sent over the Internet, sending a database may eventually overwhelm the available bandwidth in resource-constrained environments.
- *What happens when data are resent–are they ignored or do they overwrite existing data?* For example, if the user accidentally sends an older copy of the database or transmission file, when the data are received at the other end, how does the software note the age of the data?

- *How do users check the status of transmissions?* If the software includes routine data exchange, then some users will be responsible for monitoring the status of data transmissions. How will they monitor transmissions, and will they take any follow-up actions that require support from the software?
- *How reliable is the proposed communications medium?* The Internet is the medium of choice for long-distance electronic communication. Yet this communications medium often relies on the high-quality services of internet service providers (ISPs). It also relies on other components of the communications infrastructure such as telephone lines. Before designing a computerized LMIS based on electronic exchange of data, project participants should consider whether the proposed communications medium is sufficiently reliable to support the routine transmission of LMIS data.

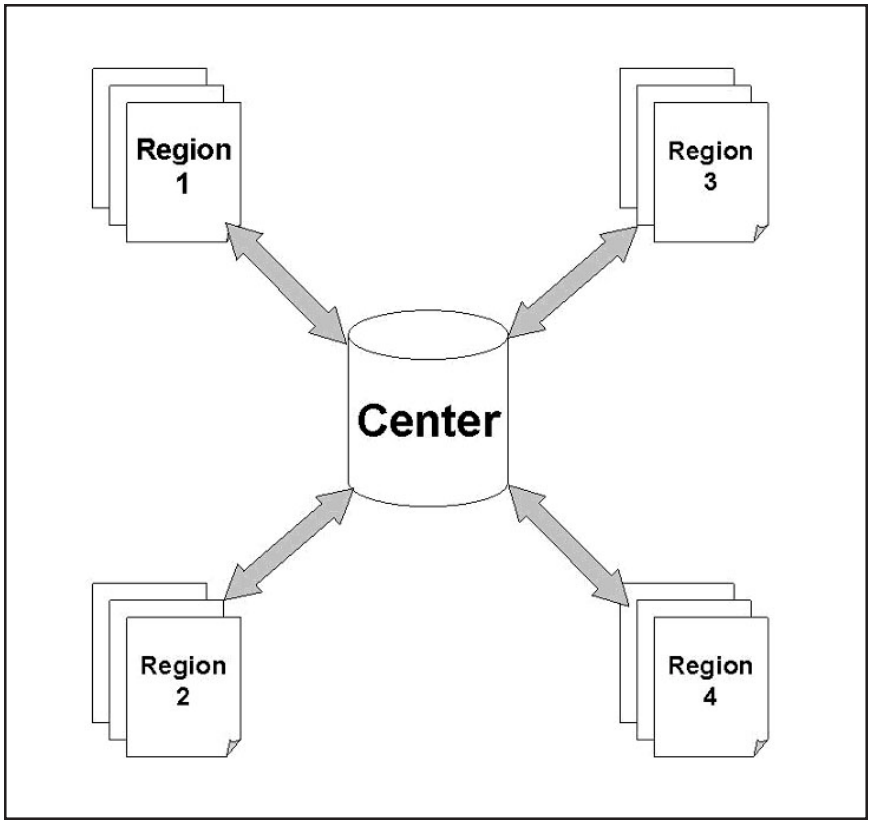
Database Configuration Options

There are basically two options for configuring a database to enable electronic sharing of LMIS data among distribution points within the logistics system: install a separate copy of the database at the distribution point (distributed database) or allow each distribution point to access a central database via the Internet (central online database). A third database configuration, implemented by JSI in several countries, is to install a database in one central location for collection of data from paper reports (central database). This third option does not automatically allow electronic sharing of LMIS data but, technically speaking, is the easiest to implement.

Central Database

JSI has used a central database configuration to develop and implement a computerized LMIS in several countries. (See appendix 1 for lessons learned in implementing computerized LMIS in five countries.) In this configuration, the computerized component resides in a central processing location—typically, the procurement and logistics unit of the MOH—and all facilities submit paper-based reports to the central location. Figure 1 depicts the configuration of a central database.

Figure 1: Central Database



Distributed Database

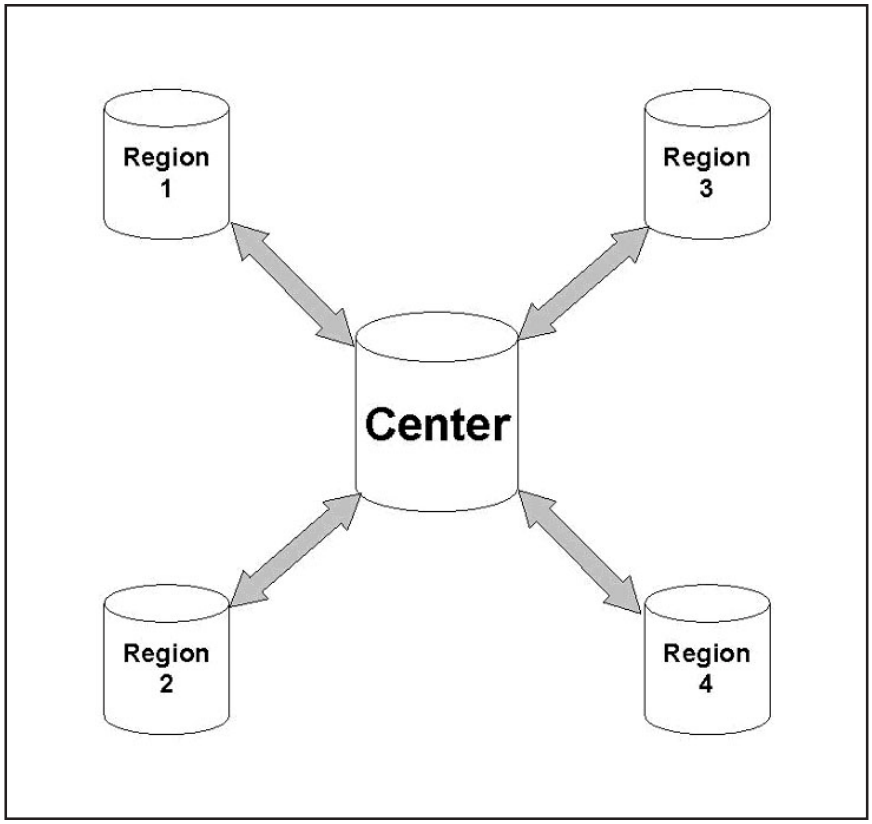
In a distributed database configuration, the computerized LMIS is installed at multiple locations in the distribution system; each location has its own copy of the software and database. In this configuration, data are exchanged electronically in place of paper-based LMIS reports.

Among distributed databases, LMIS reports can be exchanged in several ways—

- Attached to an email sent via the Internet.
- Copied onto floppy disk sent via postal mail.
- Sent to the central database via the Internet using a built-in database synchronization utility.

A distributed database configuration is the most difficult to design, develop, and test because it relies on an exchange of electronic LMIS reports between separate databases. Before developing and implementing a computerized LMIS based on distributed databases, project participants must address the questions listed in Electonic Exchange of LMIS Reports. Figure 2 depicts the configuration of distributed databases.

Figure 2: Distributed Databases



Central Online Database

A central online database configuration offers the best of both worlds—the technical ease of a single database combined with the ability to make logistics information immediately accessible to managers who make operational decisions at lower levels of the distribution system. In this configuration, the computerized LMIS and database are accessible to lower levels via the World Wide Web. Lower levels use the computerized LMIS to enter, update, and delete logistics data, but their changes are recorded in a central database.

The primary disadvantage of a central online database configuration is that it relies on a fast and reliable Internet connection at each location that accesses the computerized LMIS over the Web. Reliable Internet capability is generally not feasible in resource-constrained environments. Figure 3 depicts the configuration of distributed databases.

Figure 3: Central Online Database

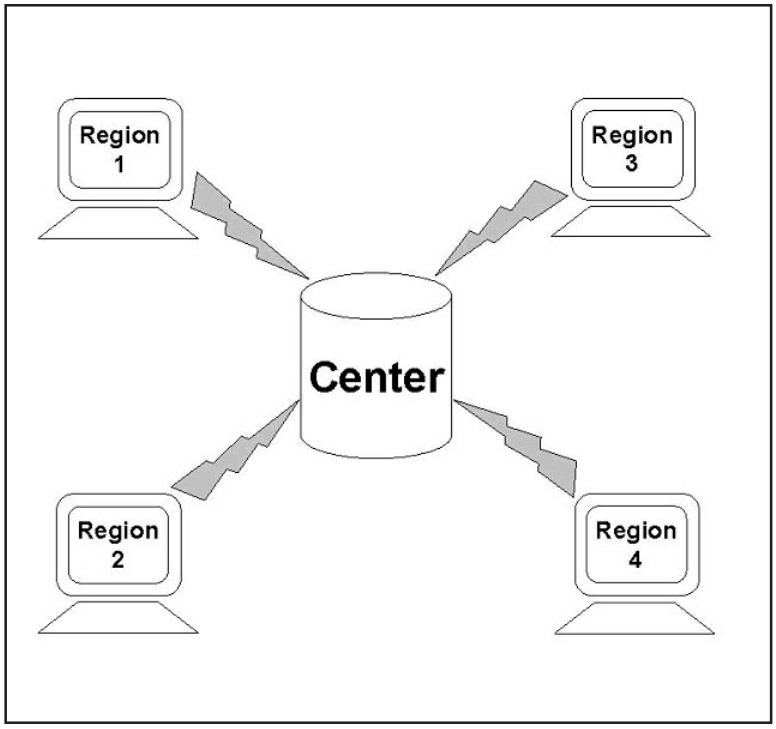


Table 8 lists the advantages and disadvantages of each configuration option.

Table 8: LMIS Database Configuration Options

	Central Database	Distributed Databases	Central Online Database
Advantages	▪ Easier and less costly to support one database as opposed to 10 or 20 databases. ▪ Organization can concentrate skilled and trained staff at central level. ▪ Data transfer is simple and not prone to technical glitches.	▪ Mid-level facilities relieved from manually aggregating data and performing calculations. ▪ Facilitates monitoring (non-reporting, stockouts, or overstocks) at lower levels. ▪ May facilitate more timely data entry.	▪ Best of both worlds: lower levels have access to computer system (benefits of distributed system), but system is maintained centrally, and user doesn't need to worry about data transfer (benefits of central system).
Disadvantages	▪ Difficult to provide timely information for operational decision making.	▪ The ongoing challenge of keeping distributed databases synchronized. ▪ Need to define procedures for updating data—for example, what happens when previously submitted data are changed? ▪ At lower levels, constraints in human and technical resources increase the complexity and cost of training and technical support.	▪ Requires reliable and reasonably fast Internet connection; may not yet be feasible in many resource-constrained countries.

DEVELOPING THE LMIS

After project participants have agreed to a design for the computerized LMIS, the design is handed over to one or more programmers to begin writing software code. The systematic approach to development described in this section will enable development of the computerized LMIS according to the original design and in a reasonable time. A computerized LMIS developed in this way will be easier to use, enhance, and maintain. Ease of maintenance is important because in many cases someone other than the original programmer will maintain the computerized LMIS during its lifetime.

For any programming language or environment, a systematic development process includes the following activities—

- Applying naming conventions to software objects.
- Applying coding standards to software code.
- Controlling the software code.

Applying Naming Conventions to Software Objects

Using naming conventions simplifies coding, code review, code re-use, and maintenance. Depending on the programming language and database platform, programmers should adopt a naming convention for all the software objects listed in table 9. It is best to adopt the most popular naming conventions available for a particular programming language and database. The industry standard naming conventions for several programming languages used by JSI are described here.

There are a number of naming conventions available for different programming environments. Hungarian notation is the precursor of many type-based naming conventions originally developed for C and C++ and later adopted for use in Visual Basic and Microsoft Access programming environments. JSI uses Leszynski/Reddick naming convention, a popular variant of Hungarian notation, for Visual Basic and Access Basic programming environments. According to the Leszynski/Reddick convention, syntax for each software object should follow this format:

[prefixes]tag[Basename][Qualifier]

For example, a product name in the database would be tblProduct.strProductName.

More information on the Leszynski/Reddick convention is available at http://www.xoc.net/downloads.

For development in Oracle, JSI has created an in-house naming convention. For example a canvas in Oracle Forms would be named as SHIPMENT_CV. This naming convention is outlined in the technical documentation for NEWVERN, an Oracle application that manages central procurement and shipping of USAID-funded contraceptives.

For Java programming projects, JSI uses Java naming conventions, which are available at http://java.sun.com. The Java naming convention focuses on a few simple case rules to distinguish the functions of identifiers—packages, classes, methods, variables, and constants. Packages are named as the organization's domain name in reverse. As an example, a utility program for computerized LMIS developed by JSI would be named com.jsi.lmis.util. This approach provides a globally unique identifier for each software object, making the object easier to identify and reference in code.

Table 9: Software Objects

Software Object Type	Software Objects
Application Objects	▪ Main LMIS application ▪ Application modules ▪ Application submodules
Database Objects	▪ Permanent tables ▪ Temporary tables ▪ Columns ▪ Stored procedures ▪ Sequences ▪ Indexes ▪ Constraints
Presentation Objects	▪ Forms ▪ Reports ▪ Canvas
Display Objects	▪ Widgets ▪ Form items
Program Objects	▪ Libraries ▪ Program units ▪ Record groups ▪ Blocks ▪ Queries ▪ Cursor ▪ Local variables ▪ Global variables ▪ Passed parameters ▪ Received parameters

Applying Coding Standards to Software Code

Coding standards help ensure that the software code appears visually consistent, can be easily read and understood by other programmers, and can also be read by tools such as *javadoc* to automatically generate program documentation.

Coding standards typically cover the following areas:

- syntax and placement of comments
- indentation
- commands from your program variable and literal.
- differentiation of global and local variables
- code structure, including–
  - declarations
  - code body
  - exception handling
  - exit points
  - division of code into logical units
  - function return values
  - error handling.

Popular naming conventions discussed in this section address some of these coding standards. Best practices, coding guidelines, and sample codes can be found in Microsoft MSDN, Sun's Java websites, W3W consortium, Apache website, coldfusion communities, and open source communities.

A common best practice is to write template code based on coding standard best practices for a particular programming language and platform. You may want to write several templates that represent your typical programming scenarios. For the NEWVERN application, JSI has developed a number of templates for typical forms and reports.

When developing template code, developers should plan to conduct code reviews to ensure that the code rigorously follows the naming convention and coding standards; otherwise developers may inadvertently introduce coding standards that are non-standard.

Another best practice is conducting periodic code reviews with experienced programmers. During code reviews, several developers should jointly review the code for the quality of the logic and the consistent application of naming convention and coding standards. These reviews provide several benefits: developers learn from one another; more efficient coding algorithms are developed under experienced guidance; and more than one developer becomes familiar with the code, thus facilitating maintenance.

Controlling the Software Code

Some large computerized LMIS projects may require the efforts of multiple developers. And, after first implementation, all computerized LMIS—large and small—will undergo various bug fixes, patches, and versions. Both situations require a source code control system to prevent inconsistencies in the software code that arise when many developers are contributing code or when the code is being changed over time. Choice of a source code control system often depends on the particular database and/or programming language that the computerized LMIS is based on.

A source control system stores the code base in a central repository. It lets individual developers check out source code files to work on. Developers can lock files at the repository to prevent others from working on the same file at the same time. When unlocked files are being worked on by multiple developers, the repository applies rules track which changes were made by which developer.

Source control systems also enable version management. Developers can manage different versions of the software code by naming them version 1.0, 1.1, etc.

For the Microsoft programming environment, Visual Source Safe is a popular choice. For Oracle environment Oracle Software Configuration Manager (SCM) is a common option because it handles binary file formats used by several Oracle development tools. Various third-party source control systems are also available for the Oracle environment. For open source and Linux-based software development, Concurrent Versions System (CVS) is a popular option.

BUYING THE LMIS

For clients, there are two alternatives to developing a computerized LMIS: buying packaged, off-the-shelf software (like word processing software) or buying the services of an information technology consultant or organization. Buying packaged software is an attractive alternative for implementing computerized systems that have common features across multiple organizations and environments. Hiring an external consultant or organization may likewise be a suitable approach for clients with limited human resources or information technology skills.

Buying Packaged Off-the-Shelf Software

In general, it is better in general to buy packaged software than to develop software, for these reasons—

- *Lower maintenance costs over the long term.* While the initial cost for licensing and technical support may appear daunting, the long-term costs for maintaining packaged software are lower than the costs for custom-developed software. The cost for maintaining packaged software is spread across a large number of clients, so each client's share of that cost is small compared to the cost for maintaining software developed specifically for one client.
- *Faster implementations.* Packaged software has already been tested by the vendor as well as by numerous clients, so unlike custom-developed software, it does not need to be tested thoroughly prior to implementation. It was also designed to be installed in a wide variety of client environments, and so it includes utilities to facilitate installation.
- *Access to upgraded versions of the software.* Vendors continuously develop new versions of packaged software to fix identified defects and to include suggested enhancements from clients. Clients can then choose to upgrade to the latest version of the packaged software rather than spend time and resources developing custom modifications.
- *Access to product and user support.* When clients buy a software package, they also have the option of purchasing support from the vendor for some period of time, usually one year. This support helps address any technical problems clients experience using the software. In addition to support agreements with the vendor, clients can find answers to their questions from other users of the software, often via internet-based discussion forums set up for that purpose.

Although all of these are powerful incentives to buy rather than develop software, all computerized LMIS implemented by JSI have been custom-developed and not purchased. Development of a computerized LMIS to meet the particular requirements of each distribution system has been JSI's approach in several countries, for two reasons: (1) the sum information requirements of each distribution system are unique; and (2) the cost to customize packaged software to meet these unique requirements—often in excess of U.S.$100,000—would exceed the funding available to most public health organizations.

Steps in Buying Packaged Software

Assuming that sufficient funding is available to support purchase and customization of packaged off-the-shelf software, the process for identifying and selecting a package includes the following steps—

1. *Form a selection committee.* The selection committee should include the product manager, representatives of the client organization. Committee members will identify and evaluate candidate packages and select one package for implementation.

2. *Create an evaluation checklist.* Based on the requirements identified during analysis of client needs (see pages 15 to 18), the product manager or other committee members prepare a checklist or scorecard for evaluating various packages. The checklist should include the following categories—
   - Functions that the software must perform.
   - Outputs (reports and graphs) that the software must produce.
   - Hardware, operating systems, other software, and network that the software must work on or with.
   - Number of users to be supported.
   - Deployment plan required (installation, setup, testing, data migration, and training).
   - Special requirements (language, accessibility, and interface with other soft ware).
   - Total cost for purchase and installation.
   - Post sales/installation support (service, response time, and cost).

   For each category, the selection committee should specify the minimum acceptable criteria, additional desirable criteria, and scores for both the minimum and desirable criteria. In addition, the committee should identify which criteria are essential so that any package that fails to meet essential criteria will not be a candidate for selection.

3. *Determine how suitable candidate packages will be identified.* The selection committee can choose to identify candidate packages in one of three ways—
   - Shop on the Internet, in catalogs, or in industry literature.
   - *Prequalification*, in which a notice is published (usually in one or more newspapers) for companies to express interest in and present their qualifications to bid on the proposed system. The selection committee will evaluate all bids received and select a small group of companies to bid on the proposed system.
   - *Open tender*, in which system specifications are published and companies are invited to present proposals for the desired system.

   If the proposed system is small or the functions are standard (such as those listed on page 13), then shopping around may be the best option. The larger or more complex the proposed system, the more likely the committee will need to use prequalification or open tender.

4. *Evaluate candidate packages.* After the committee has selected an approach to identifying candidate packages, committee members evaluate candidate packages using the checklist created earlier. Three or four companies with the highest scores are then notified that they remain candidates for selection, while the other companies are notified that they are no longer candidates.

5. *Evaluate the top three or four packages through demonstrations.* The selection committee invites the top three or four companies to demonstrate their package. The demonstration should include real logistics data from the client so the client should plan to provide these data to each company beforehand.

6. *Select a package.* Following the demonstrations, the selection committee selects the package whose functions and user interface best meet the requirements in the evaluation checklist.
7. *Develop a contract for purchase, customization, installation, and training.* Developing a sound contract is one of the most important activities in buying a computerized LMIS. The contract should specify expected contract deliverables (which may include development process documents like the samples in Appendix 3); standards for coding hardware and software; and mechanisms for monitoring progress and managing requested changes in requirements.

Buying the Services of an IT Consultant or Organization

Another way to implement a computerized LMIS is to purchase the services of an outside consultant or company. JSI has used this approach in several countries: Bangladesh, Nepal, and the Philippines. (See appendix 3 for lessons learned in implementing computerized LMIS in these countries.) This can be an effective approach to computerized LMIS implementation in public health organizations that have limited or no information technology skills. It also allows these organizations to benefit from the skills of outside consultants or companies who have experience implementing computerized systems for a number of clients in a variety of settings.

TESTING THE LMIS

Testing software before it is put to use is a critical activity in the process of developing and implementing a computerized LMIS, but is often ignored. In many cases, testing is done informally by the developer as each function of the LMIS is developed or enhanced. This informal approach is not adequate to ensure that the LMIS meets the client's objectives. To ensure software quality, a more formal and rigorous approach to testing is required.

During testing, testers focus on verifying that the software does what it is designed to do. Other major goals of testing include the following:

- To uncover defects (bugs) in the software.
- To make sure the software doesn't do what it is not supposed to do.
- To have confidence that the software performs adequately.
- To understand just how far we can push the system before it fails.
- To understand the risk involved in releasing a system to its users.

Types of Testing

There are two general types of testing techniques: positive testing and negative testing. These techniques are complementary, and both should be used in the testing process.

Positive testing focuses on the main goal of testing mentioned here—confirming that the software does what it is supposed to do. The main tool for positive testing is test cases that are based on the requirements assessment document developed during the earlier phases of the software project. A well-written requirements assessment document proves invaluable at this point, because writing test cases may be as simple as adding to the original requirements assessment document a column for the tester to record the results of the test. An example of a test case is included in the sample process documents section.

Conversely, *negative testing* aims to make sure that the software does not do what it is not supposed to do. An example of negative testing would be to verify that the software does not let the user record a negative value in a field for recording monthly stock receipts. A useful tool for negative testing is the list of business rules that the system must adhere to; this list is compiled during the design phase of the project.

As a general rule, a computerized LMIS that will be widely distributed or that will have a large number of users should undergo extensive negative testing. The reason for this is that some of these users will likely have less experience or training, and will, therefore, make mistakes with the software; the system should detect these mistakes and alert the user. Otherwise, unexpected user input may cause the system to record bad data or crash without warning. For systems with a very limited number of users, minimal negative testing may be acceptable, although it is always preferable to conduct as much negative testing as resources permit.

Before the computerized LMIS is released for use, *it is vital that someone other than the developer performs tests.* Not only does this serve as a quality check on the developer's work, but it also enables testing from the users' perspective, with the testers as surrogate users. In large software projects, a dedicated team of testers may fill this role. In smaller projects, the tester may be the product manager or the users themselves.

The testing process requires frequent communication between testers, developers, and the product manager to report and fix defects and to verify fixes. The tester identifies defects or issues and reports them to the product manager or developer. The developer then fixes or addresses the defect and reports to the tester to test the fixed version. Often, there is additional back-and-forth communication between the tester and developer before a defect is completely addressed. The need to track individual defects through a sometimes lengthy testing process makes a computerized defect tracking tool—for example, a database—especially useful. Such a tool enables testers, developers and the product manager to monitor the status of individual defects as well as testing process itself.

The last step in the testing process is *user acceptance testing*. This is performed by one or more user representatives to confirm that the software works correctly and is usable before it is formally delivered to the end users. During user acceptance testing, users try the system by performing typical tasks that will be carried out during normal usage of the software. Wherever possible, to simulate real usage, actual data should be used for testing. User acceptance testing should also include reviews of any associated documentation, such as user manuals.

Clients or user representatives have the responsibility to ensure that the software undergoes user acceptance testing before it is implemented. They also have the right to detailed information on the overall testing process: the testers, test plans, and test results. If the development team cannot provide this information, it is an indication that a systematic testing process is not in place. In that case, they may need to plan for more extensive user acceptance testing before the software can be delivered to end users.

A final note on testing: all a newly developed computerized LMIS—even the most well designed and thoroughly tested LMIS—contains defects that users will uncover during operations. After the computerized LMIS has been implemented, the mechanism for

tracking defects should remain in place; this will allow both users and developers to report and fix any defects that are uncovered.

DOCUMENTING THE LMIS

If a computerized LMIS has been well designed and developed (or bought), it may be in operation for many years after its initial installation. Future users may not have been involved in the development of the software and may not know how to perform basic tasks. Even users who did participate in development may need help with advanced tasks that they perform infrequently. At the same time, over the life of the software, users will request modifications and enhancements—especially if they are using high-quality software and want it to expand to do more tasks. The developers then hired to modify the software may not have been involved in the original development work, and will need to learn the software's underpinnings—the code, database, and overall structure—before beginning any modifications. This is where documentation comes in handy. Documentation comes in several forms, including online help files, user's guides, and technical manuals.

When planning a budget to develop new software or enhance existing software, be sure to budget sufficient time and resources to produce the documentation. Documentation—particularly the user's guide—may be the first thing prospective users or sponsors see, and it should reflect the high quality of the software itself. The common approach in which programmers quickly write the documentation the day before delivery of the software is generally not acceptable; instead, the project manager should approach the software's documentation as an official publication.

Online Help

Online help files are the first place users can go to when they encounter problems or aren't sure how to do something while using the software. When users have questions, they can access the help file directly from within the software application and look for the answer or answers there. A major advantage of online help is that users can quickly find potential answers to their questions, without having to locate and search through a paper-based user's guide.

Help files generally contain two sections: the help contents (divided into books and topics) and the index. The contents outline major tasks, while the index allows users to search for and view all help topics for a particular word or phrase. The most basic help file may contain only the help contents but no index.

With well-written and up-to-date use cases, creating the help contents is a straightforward process: the contents can be based on the use cases.

User's Guide

The user's guide is a paper-based variant of the online help file. It should accompany or immediately follow the initial delivery of the software. In addition to the contents found in

the online help file, the user's guide may include minimum hardware requirements, installation instructions, and one or more tutorials. But like online help, the majority of the user's guides are detailed descriptions of the software's functions. Use cases are handy here as well—they become headings within this section and help to divide it into manageable chunks in preparation for writing.

A sample outline for a user's guide is included in appendix 3.

Technical Manual

While the online help files and user's guide are intended to help users operate the software, the technical manual helps programmers modify it. Unlike the help files and user's guide, the technical manual targets a limited audience. The manual does not need to be formatted for publication, because it is not widely distributed.

The technical manual should be written by the developers who are most familiar with the programming conventions, modules, and database structure that the manual will cover. Throughout the project, the product manager should oversee and facilitate the production of the technical manual. In many projects, cost and time overruns cause the developers to cease work as soon as the last defect is fixed, without documenting their work. Without a technical manual, future developers will probably find the software difficult to maintain and modify.

A sample outline for a technical manual is included in appendix 3.

TRAINING LMIS USERS (PERFORMANCE IMPROVEMENT)

Performance Improvement Methods

Typically, just before a new or enhanced software application is delivered, the intended users are trained how to operate it. Two major ways can be used to convey to these users, potential users, and project sponsors how to operate the software and access the data it contains:

- *Structured on-the-job training* is a planned process in which specific tasks are carried out at the work setting with the use of job aids and, in some cases, assessment tools. The instructor (usually the supervisor) guides the learner through the steps required to complete certain tasks, such as entering data, preparing to run a report, or placing an order for supplies according to data in a report. This is an appropriate method to ensure ongoing performance improvement.
- *Competency-based training* is designed to ensure that the knowledge and skills required to operate or use software (competencies) are developed under the guidance of an instructor. This type of training generally takes place in a formal classroom setting. Throughout the training session, the instructor provides participants with simulations and case studies that contain exercises meant to develop competency for specific tasks, such as entering data reported by facilities or generating summary reports. Testing of skill acquisition can be done at selected points throughout the course.

What Method Is Best for Training People to Operate the LMIS?

Every computer configuration is unique. Often, expert knowledge of the software is required to configure it to work on a particular computer. In addition, the number of people to be trained to operate the software is usually small—at most two people at each installation. Taken together, these imply that the most suitable type of training on software operations may be on-the-job training provided by a member of the development team or by a person specially trained by the development team.

However, JSI's experience implementing software has shown a major weakness in applying this as the only approach to training. It neglects to demonstrate to end users—program managers, officials, donors and policymakers—how the software's data can be used to guide and support key decisions on product selection, procurement, resource allocation, and financing. The result is that the software operates where it has been installed, but is not used to influence major logistics decisions. In 2002, JSI conducted a review of implementations of an LMIS in two African countries, which concluded that "there needs to be a focus on the use of data by logistics managers and policymakers early in the implementation process, in order to create demand for system outputs and to generate support for system operations." The next section of this chapter will discuss a possible way to bring the software and its outputs to the attention of major end users.

Competency-based training has proven effective in developing essential skills in classroom settings, and it is worth considering how to incorporate competency-testing exercises into on-the-job training. One option would be to create exercises that test the learner's ability to

(1) navigate the application, (2) enter typical data, and (3) find data that answer a particular question.

What Method Is Best for Training People to Use LMIS Data?

As stated earlier, instructing software operators to enter and retrieve data is not a sufficient training approach. To be successful, a training approach must also focus on demonstrating to end users how to use the software when making key logistics decisions. Yet, in many cases, end users do not have the time or inclination to attend a traditional, competency-based training session in a classroom. A viable training approach for these end users is to present outputs of the software's data—reports, charts, and graphs—at regular meetings or forums in which key logistics decisions are being made. These meetings will vary by country, but may include quarterly (or annual) meetings of the logistics coordinating committee, donor coordination meetings, and gatherings of high-level officials to review or change national policies governing health care.

The logistics decisions typically made at such meetings are based on the following questions—

- How much of a particular product do we have nationwide?
- How much of a particular product is used on average per month/quarter/year?
- Are we going to stock out of a particular product? When?
- How much of a particular product should we procure?

A well-designed LMIS can help answer all these questions. Such an LMIS contains the data needed to calculate the answers, although it may be required to develop reports, graphs, or other outputs tailored to the information needs of a particular distribution system. New or customized reports are generally modest changes to the software, and the software should provide the capability to create them as needed.

Table 10: Two-pronged Training Method for a Computerized LMIS

Training Topic	Type of Client	
	Software Operators	Decision Makers
Configuring the software	✓	
Entering background data	✓	
Entering transaction data	✓	
Generating reports	✓	
Requesting reports		✓
Monitoring stock status		✓
Determining quantities of supplies to procure		✓
Supervising employees		✓

DEPLOYING THE LMIS

Many software applications end up being operated on multiple and diverse computers. Software that was originally developed to meet one client's specific need can be useful to other clients with similar requirements. For these reasons, planning a smooth deployment process is an important element in the ongoing success of a computerized LMIS. Installation and setup routines that are simple and easy to follow not only facilitate implementation of the software on multiple computers but may also make the software a more attractive solution to future clients.

Effective deployment of a computerized LMIS includes the following two activities:

- creating installation files
- creating packaging.

Creating Installation Files

A poor installation process can quickly kill the promise of a new computerized LMIS: encountering numerous, unreadable error messages, or discovering that files critical to the installation are missing, may cause clients to give up on the LMIS altogether. A more effective installation process is based on an installation tool that presents a wizard-like interface to guide the client through the installation process. Two popular installation tools are InstallShield (www.installshield.com) and Wise (www.wisesolutions.com).

Creating Packaging

High-quality packaging can demonstrate to potential future clients that the computerized LMIS is of similar high quality. Packaging includes the following elements—

- *User's manual format.* A user's manual is a must for any computerized LMIS. A user's manual in an easy-to-read format makes the computerized LMIS more understandable to both current and future clients.
- *CD or diskette label.* It is worth purchasing blank CD or diskette labels and writing the name of the computerized LMIS on them. The CD or diskette label immediately identifies the LMIS to clients and will prevent the CD or diskette from becoming lost.
- *Informational brochure.* A one-or two-page brochure advertising the computerized LMIS is optional, but may be useful for an LMIS that can be applied to numerous client environments.

MODIFYING THE LMIS

Most computerized LMIS implementation activities deal with the ongoing and never-ending tasks of fixing and enhancing existing applications. Since supporting an existing application is ongoing, project participants need a tool to continuously record software defects or suggested enhancements, to assign them to upcoming versions of the software, and to track their status during testing. Such a tool is usually in the form of a database that is always accessible to all project participants, particularly to the product manager, end users, and developers. (The section on testing software later in this document also briefly describes the use of an issue tracking tool during testing.)

Tracking Defects and Enhancements

After a computerized LMIS is operational, it's continued successful operation depends on the regular collection and tracking of two types of software issues—

- *Defects* are faults in the software that prevent or hinder users from efficiently performing their tasks;
- *Enhancements* are suggested by users or the product manager to improve or expand software performance, often in response to a change in the business environment.

Both types of issues can be tracked in the same database, but may require very different responses from the developers, depending on the severity of the issue reported. Some defects may need to be fixed as soon as they are reported—especially any critical defects that prevent users from performing an essential task—but others can be grouped with the enhancements awaiting development of the next version of the software. In the latter case, following the steps listed later in this section can help to ensure that new versions of the software are released on schedule and within budget and also contain all the enhancements and defect fixes that users expect.

Recommended Components of a Defect and Enhancement Tracking Tool

The tool used for tracking defects and enhancements should, at a minimum, track the data items and generate the reports listed in table 11. The severity data item describes the impact of an issue, as outlined in table 12. The list of reports should provide basic support to the product manager and other project participants in planning and managing new releases of existing computerized LMISs.

A defect and enhancement tracking tool in Microsoft Access (the Incident Database) was developed by JSI several years ago, and was used to record issues for all software applications. Some JSI software developers are now telecommuting, and this has required the use of a tool that both onsite and offsite project participants can access and update. JSI has recently implemented Bugzilla, an open source, Web-based defect tracking tool that tracks defects and enhancements for software applications developed by JSI in the United States.

For larger or more complex enhancements, JSI has also developed a paper enhancement specification form. A sample enhancement specification form is included in appendix 3.

Initial Steps in Modifying a Computerized LMIS

- *Establish a target release date.* The product manager establishes a target release date for the next version, according to the schedule of activities in the work plan (if it exists).
- *Review and prioritize the list of defects and enhancements.* The product manager reviews all enhancements and defects currently listed in the issue tracking tool, and closes any that are no longer relevant or that do not apply. In consultation with users and project sponsors, the product manager prioritizes the list for inclusion in the next software version.
- *Estimate time required to make requested changes.* The product manager gives the prioritized list to the developer, who then estimates the time required to make all the changes requested.

- *Finalize the list of enhancements for the next version.* The product manager finalizes the list of enhancements based on the existing budget and the estimated developer effort. The product manager may then defer some proposed changes until later versions, if the developer effort exceeds the current budget for releasing a new version.
- *Establish a preliminary schedule for releasing the next version.* The product manager develops a preliminary schedule based on estimated developer effort, the target delivery date, and availability of the developer and testers.

Table 11: Data Items and Reports in a Defect and Enhancement Tracking Tool

Data Item	Reports
▪ Person reporting the issue ▪ Date the issue was reported ▪ Application and module where the issue was encountered (if you support more than one application) ▪ Severity of the issue (see table 12) ▪ Priority of the issue ▪ Summary description of the issue ▪ Detailed description of the issue ▪ Target software version ▪ Person assigned to address or resolve the issue ▪ Date the issue was addressed or resolved ▪ Description of the resolution ▪ Date the resolution was verified	▪ Open (unresolved or unverified) issues by application and module ▪ Closed (resolved and verified) issues by application and module ▪ Issues by severity ▪ Issues by priority ▪ Issues by target version

Table 12: Types of Severity of Software Defects

Severity	Description
Blocker	Blocks development and/or testing work.
Critical	Software crashes, loss of data, severe memory leak.
Major	Major loss of function.
Normal	Average bug.
Minor	Minor loss of function or other problem where an easy workaround is present.
Trivial	Cosmetic problem such as misspelled words or misaligned text.
Enhancement	Request for enhancement.

COMPUTERIZED LMIS OPERATIONS

Efforts to design, develop, and implement a computerized LMIS may take months or even years and may involve many people. But, the hard work doesn't end after the LMIS has been implemented. Successful ongoing operation of an LMIS requires several activities—placing the LMIS within the central organization, managing LMIS data; monitoring data quality, and providing technical support. These activities are described in more detail in this chapter.

PLACING THE COMPUTERIZED LMIS WITHIN THE CENTRAL ORGANIZATION

There must be one organizational unit at the central level in which the computerized logistics information system is physically and administratively located. The logistics information system could be located in one of three possible organizational units—

- The department that has responsibility for procurement, storage, and distribution for all pharmaceutical and medical supplies.
- The department with responsibility for information technology and management information systems (IT/MIS).
- The programmatic department responsible for delivery of the health services that will be used by the health supplies.

There are advantages and disadvantages associated with placing the logistics information system in each of these units. The procurement and logistics unit is the first and natural choice for several reasons. First, it is the source of most of the data on storage and distribution of supplies. Second, it is the primary user of the information generated, which is the basis for deciding when, where, and how much to ship. But, a major disadvantage of locating the system in the logistics unit is that logistics personnel may not have the information technology skills and resources required to operate and maintain a computerized system.

Conversely, locating the system in the IT/MIS unit can ensure that both hardware and software that make up the system will receive adequate technical support. Placing the system within this unit also facilitates integration or comparison of logistics data with data on service delivery or other health indicators. At the same time, giving the IT/MIS unit primary responsibility for the system leads to a major disadvantage: the users of the data in the system—logistics managers and managers of various health service delivery programs—may not be able to access the data in time to make key decisions affecting supply. In that case, the logistics information system loses its main usefulness.

Housing the logistics information system within a health service delivery program confers a major advantage: these programs are the ultimate clients of the logistics system and, therefore, the ultimate beneficiaries of high-quality logistics information. They are responsible for the rational use of supplies, and they rely in part on logistics data to monitor and evaluate the effectiveness of their work. But, like personnel in the procurement and logistics unit, program personnel may lack the technical skills to maintain a computer-based

system. In addition, programs may focus on more high-profile, donor-driven health information systems (HIS), and neglect the logistics information system.

The precise location of the logistics information system within a program unit may depend on whether or not the logistics function is integrated across all programs or is managed vertically by program. If the logistics information system is, or is likely to become, integrated across some or all programs, it should be placed high enough within the organization to represent all the programs it will serve.

Ultimately, organizational politics may dictate the location of the logistics information system without much weight given to these technical considerations. Nevertheless, it is important that the advantages and disadvantages be considered in order. For example, to anticipate problems that might arise due to the particular location of the system within the organization, if the system is to be located within the IT/MIS unit steps must be taken to ensure the timely flow of data into the system, as well as ensure that both logistics managers and program managers have timely access to information in the system.

Table 13: Advantages and Disadvantages of Location of the Computerized LMIS

Organizational Unit	Advantages	Disadvantages
Procurement and logistics unit	▪ Original source of data on inventory and distribution ▪ Primary user of logistics information system data	▪ Personnel may not have information technology skills or resources ▪ Low priority of procurement and logistics function within the organization may lead to low visibility of the LMIS.
Information technology/ management information systems unit	▪ Personnel have information technology skills. ▪ Facilitates integration or comparison with other sets of information managed by the unit (for example, service statistics).	▪ Logistics managers and program managers may not have timely access to logistics data for decision making.
Program unit	▪ Personnel gain direct access to logistics data to monitor and evaluate effectiveness of their programs. ▪ Personnel may not have information technology skills or resources. ▪ Logistics information system may receive little attention compared to higher-profile HMIS.	

MANAGING COMPUTERIZED LMIS DATA

Operation of a computerized LMIS includes the following key tasks—

- *Collecting, entering and validating routine LMIS data.* A successful LMIS relies on routine collection and processing of LMIS data received from facilities in the distribution system. It also relies on routine monitoring of receipt of data from each facility and procedures to resolve erroneous, incomplete, or late reporting by facilities.
- *Distributing routine LMIS reports.* In addition to data entry, LMIS procedures should include the routine distribution of logistics reports to provide feedback or to enable decision making. Together with data collection and entry, this task is the most important ingredient in a successful computerized LMIS. (See page 9 for a complete list of recommended LMIS reports.)
- *Responding to special requests for LMIS data.* Occasionally, LMIS users or other organizations may request LMIS reports that don't currently exist in the computerized LMIS. Procedures should be in place for handling these requests in a timely manner. Technical support personnel may need to assist with this task in some cases—for example, by developing special reports tailored to meet a particular request.
- *Identifying and reporting software defects.* Like all software, LMIS software will contain defects. A key task in LMIS operations is identifying and reporting these defects as they occur. Depending on its severity, the defect may be fixed immediately or added to the list of modifications for the next version of the software. (For more information on tracking software defects, see pages 40-41.)
- *Identifying improvements for the next version of the software.* Like all software, LMIS software will periodically need to be improved or modified in response to changing requirements. LMIS operations must include a way to document suggested improvements and modifications as ideas for improvements arise. Operational procedures must also include a way to specify the suggested improvements in more detail before developing the next version of the software. As an example, special requests for LMIS data may join the list of improvements if they are requested repeatedly by users. (See pages 40-41 for more information on identifying improvements.)

Monitoring the Quality of LMIS Data

Despite the expression *garbage in–garbage out*, there is a universal tendency to accept as true anything that comes out of a computer. It is common to overlook the actual quality of data being collected. By the time data is entered into a computer, it is often too late to control many problems with data quality.

The most valuable data in a computerized LMIS—particularly data on consumption and availability of products to clients (stock on hand)—originates at service delivery points (SDPs). Since SDPs are unlikely to have computers, there will be at least one degree of separation between data generation and data entry if the district level is computerized, and two or more degrees of separation if the LMIS is computerized at the central level only. Because many, if not most, problems of data accuracy and validity occur at the point of data collection, data quality monitoring and control must focus on SDPs.

If SDPs view the collection of LMIS data as nothing more than a bureaucratic requirement, they will give less care and attention to that task. In contrast, if LMIS data collection is designed and carried out in a way that supports decision making at the local level, then

SDPs will pay more attention to the accuracy of the data they collect and report. Otherwise, the quality of data that reaches the computerized component of the LMIS may be compromised.

A well-designed program of training and supervision, combined with an LMIS design that emphasizes the use of data at the point of data collection, will help minimize problems of data accuracy and validity. Routine monitoring of data quality—for example, monitoring indicators such as the number of errors detected in computerized validity checks, and timeliness of reporting by facilities, may also help increase the accuracy of LMIS data.

PROVIDING TECHNICAL SUPPORT

Effective computerized LMIS operations rely not only on routine data collection and reporting, but also on *technical support*—backing up the database to ensure continuous smooth operation of the computerized LMIS, as well as fixing defects and making enhancements. Technical support comprises the following key tasks—

- *Backing up the database.* When the hard drive containing the database for a computerized LMIS crashes, the data may be lost forever. One of the most important elements of technical support is ensuring that LMIS data are backed up regularly—perhaps daily—to avoid losing data as a result of hard drive failure. Technical support providers should plan to regularly back up LMIS data on an external storage medium, such as a zip disk or tape drive.

  Forgetting to back up data routinely is a common mistake when implementing a computerized LMIS. During implementation, clients and other project participants should look specifically at mechanisms and procedures for backing up LMIS data.
- *Managing user access.* Computerized LMIS, like other information systems, may assign roles and privileges to different users as a way of protecting the integrity of LMIS data. Some users might only enter and update data, others might only view reports, and one or two users may have rights to do anything (*administrative* rights). Technical support personnel should put in procedures in place for adding or inactivating users and assigning roles to these users. In some computerized LMIS, a user with administrative rights may manage user access.
- *Fixing software defects.* When LMIS users identify and report software defects, it is the responsibility of technical support personnel to fix these defects, depending on the severity of each defect—immediately, in the case of a severe defect, or during the next scheduled round of enhancements and modifications.
- *Making enhancements and modifications.* Technical support personnel are also responsible for making enhancements and modifications reported by LMIS users. These are typically gathered into a list to be included in the next version of the software.

See pages 40-41 for more information on modifying the LMIS, and pages 33-34 for information on testing.

GLOSSARY

atomic A characteristic of a column in a database table, signifying that the column contains only one value. Good database design strives to make all database columns atomic. An example of a non-atomic column is *Products Distributed*, which might contain multiple values such as *amoxycillin, cotrimoxazole, paracetamol*.

bug A defect in software that prevents the software from behaving or appearing as expected. Bugs range in severity from *blocker* (prevents continued development and/or testing of the software) to *trivial* (cosmetic problem such as misspelled words or misaligned text). (See table 12 for a list of types of severity of software defects.)

business rules Critical behind-the-scenes rules that a computerized LMIS must follow, such as adherence to a specified list of products managed by the central medical stores.

code A system of symbols and rules used to represent instructions to a computer. A major activity in the development of a computerized LMIS is writing code that instructs the computer how to store, present, and manipulate logistics data.

column A data element contained in a database table. Also known as an *attribute*.

computerized Describes an information system that is partly or wholly based on computers. Use instead of *automated*.

development The activity of writing code for a computerized LMIS.

fidelity The degree to which a user interface prototype resembles the software that will eventually be implemented.

implementation The activity of putting a computerized LMIS into first operation.

information system Any system used to collect, aggregate, and report information for decision making. An information system may be completely manual, partly computerized, or wholly computerized.

non-key column Any column in a database table that is not part of the primary key.

primary key A column in a database table that uniquely identifies each record in the table, so that information in the table can be found quickly and accurately. A primary key may be a single value, such as the product code, or a combination of values, such as the facility code plus the reporting date. In a well-designed database, all tables have a primary key.

prototype A model of the user inteface of the proposed computerized LMIS, showing all software elements that the user interacts with.

requirements	Descriptions of how an information system should behave or of a necessary system characteristic. Participants in software development or implementation attempt to gather the requirements as completely as possible at the beginning of a project to implement a computerized LMIS.
site map	A visual or textually organized model of an information system's content that allows the users to navigate through the site to find the information they are looking for, just as a traditional geographical map helps people find places they are looking for in the real world.
table	A data group in a database that describes one object in the real world, such as *facility* or *product*. Also known as an *entity*.
third normal form	A type of database in which all non-key columns in each table are mutually independent, i.e., no columns contain calculations or values that are dependent on the contents of other columns.
use case	A brief description of the interaction between a user and a computerized system to reach a particular goal. A collection of use cases represents the behavioral requirements of a computerized system (i.e., what the system is required to do to meet the user's business needs).
user interface	The elements of a computerized LMIS that are displayed to the user for interaction via a computer monitor.
wire frames	Sketches of the user interface that avoid using layout and colors but emphasize information and placeholders. These user interface-planning tools aid in the development of content and are the basis for user interface design.

Appendix 1

Lessons Learned in Implementing a Computerized LMIS

Background

Appendix 1 reviews existing computerized logistics information systems in five countries—Bangladesh, Jordan, Kenya, Nepal, and Philippines. The goal of the review was to compile a set of updated lessons—universal truths, best practices, or worst practices—that will guide DELIVER's work to advise on the development and management of computerized LMIS in more complex settings. Research methods were limited to reviews of documents and interviews of staff with mostly historical and, in some cases, current knowledge of the systems in question.

After an initial review of documents, three areas stood out as worthy of investigation: (1) organizational context and its effect on sustainability of LMIS operations; (2) mechanisms and resources for development, maintenance, and modifications of the software; and (3) utilization of LMIS data in decision making. These three areas are discussed in the following pages.

ORGANIZATIONAL CONTEXT

Organizational context refers to the agency or agencies responsible for operating the LMIS. The context of an LMIS includes the history of its organizational *home*. Organizational context also refers to the position and status of the LMIS agency within the broader host organization, other functions the agency performs, and the personnel responsible for carrying out these functions. Following are descriptions of how these contextual factors have affected LMIS operations in each of the five countries.

BANGLADESH

Development of the computerized LMIS in Bangladesh began in 1987. From 1987 to 1995, the system was housed at JSI offices and managed by JSI staff. During this period, an MOH employee— the assistant director of the MIS unit of the Directorate of Family Planning (DFP)—was seconded part-time to work with LMIS staff at JSI.

At the end of 1995, LMIS computers and operations were transferred to the MIS unit of the DFP. During the first year after the transfer, JSI seconded two staff to help ensure continuity of LMIS operations. JSI also paid for consumables (paper, diskettes, and printer toner) during this transition period.

Before the transfer took place, JSI made a deliberate decision to simplify LMIS operations. For example, six separate monthly reports were consolidated into one; processing of annual physical inventory data was eliminated; and processing of a form for tracking other health supplies—which had a low reporting rate—was also eliminated. These efforts to simplify were intended to adapt the LMIS to the reduced human resources available in the MIS unit: MIS staff were also responsible for processing and reporting on service statistics, and thus were not available to manage the LMIS full time.

In 1999, the MIS unit in the DFP was transferred to the Directorate of Health Services (DHS) and incorporated into the DHS MIS unit. This was done on the recommendation of the World Bank and other donors, who advocated the integration of previously separate functions within each directorate. Currently, the MIS unit is one of the few integrated facilities at the central level.

JORDAN

Design and implementation of a logistics information system grew out of a 1997 workshop facilitated by JSI, in which MOH and two other host country organizations redesigned the national contraceptive logistics system. The computerized JCLIS (Jordan Contraceptive Logistics Information System) was initially implemented that year at the logistics unit of the MCH Directorate (MOH). While JSI provided the technical resources to implement the system, MOH staff managed many routine system operations, including data entry and report generation. JSI also assisted the MOH in analysis of LMIS data.

KENYA

Before 1988, distribution of contraceptives fell under the purview of the MOH Medical Supplies Coordinating Unit (MSCU), which was generally responsible for warehousing and distribution of all public sector medical supplies. However, the contraceptive logistics

system was widely considered to be in poor condition, with problems throughout the supply chain. This situation was generally attributed to the low priority placed on family planning services by both service providers and high-level policymakers. To address the problem, a Logistics Management Unit (LMU) was created within the then Division of Family Health (now the Reproductive Health Division) expressly for the purpose of managing contraceptives. This established a separate, vertical logistics system for contraceptives within the MSCU.

A computerized LMIS was first implemented in the LMU in 1988. For the first several years, development and operations of the LMIS were managed predominantly by JSI and other non-MOH staff. Gradually, MOH staff were incorporated into the work of the unit. As of 2002, MOH employees manage most LMIS operations, although JSI still plays a key role in technical and management areas.

For most of its history, the LMIS has served mainly two vertical programs: family planning and, later, HIV and STI commodities. As a result, it has served a narrow set of regular clients within the MOH, but it has also maintained a focus on donors and NGOs as clients. The LMU unit itself has remained institutionally part of the Reproductive Health Division. Despite this, it continues to be heavily staffed by non-MOH employees, and there are no MOH counterparts to LMU senior mangers and technical staff.

The LMU and LMIS have been identified as potentially playing a key role in the development of an integrated logistics information system. Integration of commodity distribution has been under discussion since 1996, but has not progressed much in the past six years.

With the recent expansion of the scope of DELIVER's activities in Kenya, the computerized LMIS has been moved into an Oracle-based system designed to manage reproductive health (family planning, STI, HIV/AIDS) commodities, Tuberculosis (TB)/Leprosy products and several other pharmaceutical and non-pharmaceutical products. The system includes an integrated inventory control and distribution module and commodity-type specific forecasting and quantification modules that support the integrated use of storage, distribution, and basic inventory control forms while still generating reports and providing quantification methods specific to particular health programs. The system has also moved from the Reproductive Health Unit of the Ministry of Health to the Kenya Medical Supplies Agency (KEMSA), the new parastatal organization formed to take over the functions of the MSCU.

The focus in LMIS development has likewise shifted to capacity building of staff at KEMSA, not only to operate the computerized LMIS but also to further develop it. The goal is for KEMSA to become a logistics services solution center within the MOH, working with different units of the MOH to forecast, procure, store, manage, distribute, and deliver commodities and collecting essential data on consumption and distribution for use in decision making.

NEPAL

Efforts to develop a computerized LMIS in Nepal began in 1995. At that time, an LMIS unit within MOH was created specifically to manage the LMIS. While design and development of the LMIS was carried out by JSI, and the implementation and operation of the LMIS was contracted out to a local NGO (New ERA) for the first five years (1995–1999).

Beginning in 2000, as the New ERA contract came to an end, JSI contracted with another local company, MASS, to manage LMIS operations. Since 2000, MASS has operated the LMIS unit, with USAID funding. MOH has provided one counterpart staff member who assists with data management (reviewing and sending feedback reports). JSI continues to provide technical personnel for system maintenance, in addition to some materials and supplies.

At the time of its creation, the LMIS unit was institutionally and physically linked to the Logistics Management Division within the MOH. This institutional link has remained stable over time: since its inception, the LMIS unit has been part of the LMD, and has not been transferred to other departments. In 2002, it was situated at the highest level of the LMD organizational structure, which gives it responsibility for all logistics information functions within MOH.

One unusual feature of the Nepal LMIS is that from the outset it was an integrated system—it tracks not only contraceptives but also some essential drugs and medical supplies. This is in contrast to the other systems reviewed here, all of which started as strictly vertical systems for tracking contraceptives. Because of the wider scope of products it tracks, the Nepal LMIS potentially serves a diverse group of stakeholders within MOH and in the NGO and donor communities. The fact that the LMIS is integrated has likely enabled its high position within the logistics division of MOH.

On the other hand, there is some concern about the future viability of the LMIS unit. JSI's review of accomplishments and lessons learned in Nepal (2000) noted that "At the central level, the LMIS is threatened because of the informal nature of the LMIS unit, its staffing, and its funding." Although it appears well-positioned within LMD and enjoys a wide base of support, it relies almost entirely on donor funding.

PHILIPPINES

Development of the computerized Philippines CDLMIS began in 1991. From 1991 through 1998, CDLMIS was managed within several different departments within the DOH. Within each department, a unit for managing CDLMIS was specially created. (In other words, it was not incorporated into an existing MIS unit, as was the case in Bangladesh.) During that time, JSI provided all financial support and staffing to operate the computerized system.

Beginning in 1999, there was a gradual reduction in JSI staff assigned to help manage CDLMIS. At the same time, CDLMIS operations were transferred again within DOH, in this instance from the Family Planning Service (FPS) to the Procurement and Logistics Service (PLS). The transfer included the reassignment of five FPS staff to PLS to operate CDLMIS, although the budget to support these staff remained with FPS. JSI continued to provide eleven staff on a (part-time) basis to supplement the work.

At the time of the transfer of CDLMIS from FPS to PLS, there was some confusion about what specific tasks would be transferred. Originally, FPS planned to retain responsibility for training and monitoring, in which case it would continue to be an important consumer of CDLMIS information. Then there was a change in directors at FPS, and the new director transferred all logistics responsibilities to PLS.

During the 1990s, DOH experienced two agency-wide events that affected CDLMIS operations. The first was decentralization, in which significant decision-making responsibility—including procurement—was devolved to the 16 regions. The second was a reengineering effort at DOH to rectify a shortage in skills in the newly empowered regions. This reengineering effort resulted in major staffing changes in Manila, including the transfer of many staff trained in logistics and CDLMIS operations to other jobs within DOH. During these events, CDLMIS continued to operate as a centralized system at PLS.

By mid-2000, CDLMIS was being managed fully by DOH/PLS staff with DOH funds; JSI staff were no longer supplementing the work. During 2000, the CDLMIS staff of seven was reduced to two. As a result, CDLMIS data processing was seriously backlogged by six months or more. To determine quantities to resupply, the remaining CDLMIS staff used historical data from a year ago or more, rather than data submitted recently.

LESSONS LEARNED

- Early host organization commitment and involvement—in the form of staff, in particular—can help integrate the LMIS into the organization's operations.
  - In Bangladesh, MOH contributed the part-time services of one MIS manager when the computerized LMIS was initiated in 1987. When the LMIS was formally transferred to the government in 1995, MOH provided all operations staff and offices. As of 2002, the LMIS has become an integral part of government operations, although JSI continues to provide significant support in the form of occasional software upgrades and modifications.
  - Similarly, MOH in Jordan contributed staff to operate the computerized JCLIS. JSI provided essential technical assistance (especially in the form of software development) but, otherwise, did not play an operations role that it would have to transfer to MOH at a later stage. In addition, a design workshop helped establish host-country ownership of and commitment to the logistics system before JCLIS was created. Within a few years of its inception in 1997, JCLIS had become an essential component of MOH logistics activities.
  - In contrast, the Philippines computerized CDLMIS was developed and staffed by JSI with donor funds, for most of its operational life, so that it largely came to be perceived by DOH as a donor activity rather than an integral part of DOH operations. As a result, the CDLMIS has not been able to take firm root within the government, and, as of 2002, faces the prospect of total collapse.
- Incorporating computerized LMIS operations into an existing host organization unit, rather than a specially created unit, gives the LMIS access to established resources, and may promote host organization ownership and commitment.
  - In Bangladesh, the computerized LMIS was transferred into a well-established MIS unit of the MOH that also managed other information systems, including a system for tracking service statistics. To facilitate the transfer, the LMIS was adapted to the resources available in the MIS unit, which included the part-time contributions of experienced data entry operators and technical support personnel.
- In cases where host organization commitment or resources are lacking, the creation of a new unit within an existing institution can help the computerized LMIS get the resources it needs for initial development and operation. At the same time, additional actions or decisions may be required to promote the longer-term viability of the LMIS.

- In Kenya, the creation of a special LMIS unit staffed by JSI was followed by the gradual incorporation of MOH employees into LMIS operations. Despite this, the unit continues to rely heavily on non-MOH employees, and concerns remain about its future within MOH. However, the LMIS unit has mitigated this risk by expanding its clientele beyond MOH to include donors and NGOs—additional stakeholders with an interest in continued LMIS operations.
- Likewise in Nepal, the LMIS unit was created at JSI initiative especially for the LMIS. The unit has benefited from having the same organizational home for the duration of its existence; this has helped establish the LMIS unit within MOH. As in Kenya, the Nepal LMIS has reduced risks to sustainability by catering to a larger group of interested parties—in this case, other health units within MOH, in addition to the family planning unit.
- In the Philippines, as well, a unit was specially created to manage the computerized LMIS. But, unlike in Nepal, the unit did not have a stable organizational home; instead, it was transferred to several different DOH departments. Because of this, and also because the unit lacked high-level support within DOH, the unit was not able to become established within DOH, and continued to rely on donor support.

MECHANISMS AND RESOURCES FOR SOFTWARE MAINTENANCE AND MODIFICATION

All systems, including well-designed and well-functioning ones, need technical support—to fix defects or malfunctions in the software or hardware, or perform routine system maintenance. In addition, systems need technically skilled people who can modify, enhance, and adapt them to constantly changing environments and evolving information needs. Often, it is the successful systems —the ones that do the tasks well that they are designed to do— that come under pressure to expand, as users seek to apply that success to other tasks. Local resources are important to the process of designing and implementing modifications, as well as to more routine technical support activities.

This section describes briefly the history and current status of technical support for the LMIS in each country, as well as the process for making enhancements to the system.

BANGLADESH

The LMIS started as a spreadsheet application in 1987, and was rewritten in several different platforms over a number of years. Since 1992, the LMIS application and database have been in Oracle, which is the standard technology platform of MOH. All of these development activities were funded by USAID and carried out by JSI staff through a combination of long-term assistance and short-term visits.

After the transfer of the system to MOH in 1995, MOH staff was trained to provide basic maintenance. JSI continued to provide backup technical support, and also to plan and implement modifications to the system.

Two major modifications have occurred in the past few years. First, in 1999, to make it Y2K compliant, JSI subcontracted with a local technology firm to rewrite the entire application in a newer version of Oracle. Toward the end of the work, JSI/DC staff made a short-term visit to review the work of the subcontractor and provide guidance to JSI staff in Dhaka concerning acceptance testing of the rewritten application. This visit revealed a number of serious software defects that were corrected before the new application was installed.

Then, in 2001, the application and database were modified to accommodate the planned unification of family planning and health programs. Two JSI staff (Bangladeshi nationals) based in Dhaka carried out development and testing.

As of 2002, JSI provides resources and personnel for technical problems that MOH staff are unable to address, as well as any modifications to the system, such as the unification of family planning and health logistics data.

JORDAN

The Jordan LMIS started in 1997 as an implementation of a software application developed by JSI for projects in Thailand and Brazil. For two years, after initial implementation of this application— renamed the Jordan Contraceptive Logistics Information System, or JCLIS— MOH staff actively participated in discussions with JSI about possible enhancements to the LMIS. These discussions centered on JCLIS outputs that would be useful to logistics managers in monitoring system performance. JSI staff through short-term assistance visits then developed agreed-upon enhancements.

At one such visit in early 1999, JSI staff identified the need to hire a local firm or individual to provide technical support to the system. Several months later, JSI's resident advisor reported that "finding competent computer programming and technical support has been difficult and has slowed down the programming needs of the project." While there appeared to be competent technical support within the MOH, the MCH Directorate—where JCLIS was located—did not have the clout to access it.

During the last six months of the JSI resident advisor's tenure in Jordan, USAID funded the creation of an umbrella health project. After discussion, an informal agreement was reached stating that the new project's MIS advisor would provide on-call technical support to the MOH senior logistics officer.

KENYA

As in Jordan, the computerized LMIS in Kenya began in 1988 with an implementation of an existing software application—the CCMIS developed by CDC. After several years, it became apparent that this system had neither the functionality nor the flexibility to meet local needs. As a result, work began in 1993 on a new system developed in Clarion, which at that time was (and still is) a standard database platform within MOH. LMU staff (both JSI and MOH staff) who had first received Clarion training in the United States developed the new LMIS.

Development of the new system continued for about two years. Since about 1995, the LMIS has operated smoothly with regular maintenance and minor modifications carried out by LMU staff (JSI employees).

NEPAL

JSI and a local NGO, New ERA initially developed the computerized LMIS in 1995, with funding from USAID. By 1999 it consisted of three separate components: data entry developed in FoxPro for DOS; reporting on facility errors and on annual dispensed to user data developed in FoxPro for Windows; and feedback reporting developed in MS Access 97. In 2000, JSI hired a local consultant to integrate the three components into a single system in MS Access 97.

As of 2002, technical support is provided through a JSI bilateral project funded by USAID.

PHILIPPINES

Like the Bangladesh system, the Philippines CDLMIS evolved through a number of technology platforms after it was first developed in 1991. CDLMIS was maintained and modified by JSI staff or contractors hired by JSI for the duration of FPLM assistance (1991–2000). Donor support for the CDLMIS, through JSI, ended in 2000.

Initial development, implementation, and support activities took place without the involvement of MAS (later renamed IMS), the IT unit of the DOH responsible for supporting the DOH computer network and applications. In an effort to facilitate future support from IMS, in 1996 the application was redeveloped in Powerbuilder with a Sybase database backend, which are DOH standard technologies.

Despite this and other efforts to institutionalize technical support (including a formal agreement between USAID and DOH), the continued lack of technical support from IMS remained a major issue of concern during the last few years of FPLM assistance. In 1999, IMS belatedly assigned two programmer-analysts to support the LMIS part-time, although, at the time, JSI consultants expressed concern that this level of support would be insufficient.

In 1999, JSI hired a local technology firm (an affiliate of Sybase) to migrate CDLMIS to newer, Y2K-compliant versions of the application and database. Several months later, JSI provided short-term assistance to upgrade LMIS hardware (server and client workstations). At that time, it was expected that JSI's role would be to assist IMS in upgrading the hardware, but IMS ended up participating only at the end of the upgrade work, mainly because of a failure by PLS staff to coordinate upgrade plans with IMS.

After reports of poor LMIS functioning, JSI staff traveled to Philippines in early 2002 to assess system performance. During the visit it was noted that CDLMIS was receiving no support from IMS staff. One of the two remaining CDLMIS staff was providing basic maintenance.

LESSONS LEARNED

- Implementing an existing application and then adapting it to local requirements can enable a basic LMIS to operate from the outset, while LMIS staff gain the knowledge and experience necessary to guide the design of modifications and enhancements.
  - The Jordan LMIS was based on an application developed for projects in other countries. After implementation of this imported system in 1997, MOH staff worked with JSI over two years to identify and design software enhancements that would meet the particular needs of Jordan's logistics system.
  - The LMIS in Kenya also began with the introduction of an existing application with predefined functionality. The LMIS unit then made a single transition to another technology platform that conformed to MOH standards, after several years of identifying requirements for LMIS functions specific to Kenya.
- Compliance with local technology standards may facilitate the provision of staff by the host organization, but does not guarantee the provision of adequate technical support.
  - In Bangladesh, the conversion of the LMIS to Oracle prompted MOH to assign employees to support the LMIS, but these employees did not have the skills to fully support it. JSI has continued to provide critical technical assistance to maintain and upgrade the software.
  - In Kenya, the LMIS was redeveloped in the MOH database technology standard, Clarion, but continues to rely on technical support from JSI.
  - The Philippines CDLMIS was rewritten in the DOH technology standard within a few years of its initial development, but it has never received regular or adequate support from DOH computer support staff. In this case, the technical capacity exists in-country, but DOH decision makers are not committed to ensuring that CDLMIS can access it.

- Backup technical support can be an effective donor contribution to computerized LMIS operations that are otherwise operated and supported by the host organization.
  - In Bangladesh, Jordan, and Nepal—where the computerized LMIS is operated with host country resources—donor funding supports essential technical assistance that is either unavailable or difficult to obtain within the host organization.
- In organizations where the computer support unit is separate from the unit operating the LMIS, mechanisms that enable the LMIS unit to *buy* services from the computer support unit may help ensure technical support.
  - In the Philippines, the unit formally responsible for computer support (IMS) is institutionally and physically separate from PLS, the unit that operates the computerized CDLMIS. Such a configuration can be beneficial, because it enables DOH to pool technical resources and skills and make them available to the entire organization. But, in the absence of mechanisms for PLS to *buy* services from IMS, there is little incentive for IMS to support CDLMIS.
- Outsourcing to local technology firms may be the most feasible and efficient way to make software modifications. This enables the government to buy services from the private sector, as needed, rather than trying to compete with it for technically skilled people.
  - In Bangladesh, Nepal, and Philippines, JSI hired local technology firms or consultants to make major software modifications. This approach serves as a model for host organizations to follow for future modifications.
- Even when the computerized LMIS is managed and supported locally, short-term external assistance can continue to play a vital role through the transfer of skills in software development process management.
  - In Bangladesh, JSI provided short-term assistance to review and guide software modifications being done by a local contractor. This visit facilitated the transfer of skills in the critical area of software testing.

UTILIZATION OF LMIS DATA FOR DECISION MAKING

The ultimate measure of an information system is how the information is used to make logistics decisions—both routine, operational decisions, and longer-term strategic decisions. This section examines the use of data from computerized LMIS in each of the five countries. Also of interest are activities to address information use during the initial design process and activities (formal and informal) to promote information use during the implementation process. Both are important proactive steps to ensure that the LMIS provides useful information, and that logistics stakeholders are aware of and understand how to use this information.

BANGLADESH

The main output of the computerized LMIS in Bangladesh is the *Family Planning Monthly Logistics Report,* which is sent to family planning managers in Dhaka and to managers of central, regional, and district warehouses (21 facilities total). This report presents results by facility on stockouts and potential stockouts, reporting rates, stock status, and consumption trends. The report also ranks warehouses based on performance of months of stock on hand and provides feedback to the central, regional, and district warehouses on how they can improve performance.

At the central level, issues and stock status data are used to inform procurement decisions at donor coordination meetings. A relatively large donor community—consisting of bilateral and multilateral donors—contributes contraceptives to Bangladesh, and LMIS data has proven essential to the complex task of coordinating donor activities. The computerized LMIS enables MOH to orchestrate the components of this task—namely, determining quantities to order and planning shipment schedules.

Central, regional, and district warehouses use this report to compare their performance to other facilities and identify ways to improve their own performance or the performance of facilities they supervise. In theory, higher-level facilities can also use the report to determine quantities to issue to the facilities they supply, and to monitor stock balances.

In practice, the report arrives too late to be useful to warehouses in determining quantities to supply to lower-level facilities. To get the timely information they need, some warehouses have developed their own computerized systems for calculating issue quantities. These systems capture the same information that is processed by the central LMIS, but they are used to drive immediate supply decisions rather than longer-term procurement and shipping decisions.

JORDAN

The Jordan LMIS includes a variety of reports designed for use by the Senior Logistics Officer to assess performance of the overall logistics system, as well as the performance of individual facilities. The reports are grouped into three types: (1) administrative reports, which include a report listing non-reporting facilities; (2) graphs, which display average

countrywide stock level, CYP, dispensed to user, and service statistics (new and continuing users) by month; and (3) logistics reports, which include aggregate stock movement and facility stock movement.

KENYA

The Kenya LMIS is comprised of four modules, each of which produces outputs for operation and management of the logistics system. The *Forecast* module generates about 12 reports that include aggregate information on commodity projections, donor commitments, and order tracking from commitment through customs clearance. The *Inventory Control* module produces a variety of reports to assist in the distribution of commodities. The *Distribution Resource Planning* module produces reports for monitoring and planning distribution of commodities. Finally, the *Present* module is a tool for organizing reports from other modules into customized presentations, enabling presentation of data targeted to specific decision makers.

At the central level, LMIS data are used primarily to plan distribution of commodities to the districts. The data are also used to forecast contraceptive requirements, and to identify potential stockouts at the national level so emergency procurement orders can be placed.

At lower levels, district logistics management teams and service delivery providers have received extensive training on operating the manual components of the LMIS. It is not clear to what extent this training focused on data collection versus the utilization of information for improved decision making.

NEPAL

Outputs of the computerized LMIS in Nepal are primarily geared towards central-level managers and to a lesser extent toward managers at the regional level. The system produces several reports that enable managers to assess system operations on a quarterly basis—the reporting status and stock status of each facility in the distribution system. Another quarterly report highlights inconsistencies in submitted reports. In addition, annual reports provide dispensed to user data by district, region, or the entire country.

These reports have demonstrated significant improvements in logistics management in the country since 1995—most notably a reduction in stockouts. At the central level, they are used primarily to provide dispensed data for procurement planning of selected commodities; and to determine quantities to push to districts on an annual basis. At lower levels, the quarterly reports are sometimes used to identify stock imbalances among facilities and to redistribute commodities accordingly.

For the manual components of the system, in operation at the regional and district stores and SDPs, JSI has sponsored extensive logistics training focused on data quality and use of information for decision making.

Despite these training efforts, JSI's lessons learned report (2000) states that "within the record system, inaccuracies and errors were common and the reports were not being used for decision making." In addition, a 1998 donors' report on the state of the logistics system notes that "there have been complaints from some MOH decision makers, USAID, and other donors that the LMIS feedback reports, as currently formatted, are of limited practical use because they are too long and contain too much raw, unanalyzed data." Since

1998, annual reports produced by JSI have included summary reports and graphs on quantities dispensed of selected commodities over a five-year period; stockout rates of selected commodities also over a five-year period; and projected contraceptive needs. It is not clear to what degree these annual reports address the needs expressed by donors for more summary analyses of LMIS data.

PHILIPPINES

The Philippines CDLMIS produces two major reports designed to assist logistics operations: (1) a summary delivery report, which lists stock status and usage for each facility; and (2) a feedback report designed to alert provincial and regional family planning managers to current or potential problems in facilities they supervise—problems such as stockouts, low stocks, calculation errors, or reporting inconsistencies.

The first report has been used by CDLMIS managers to determine quantities of contraceptives to supply to provinces and cities. This determination is based on aggregated consumption reported by all the facilities within that province or city, and the stock on hand at the provincial or city warehouse. Total average monthly consumption for all facilities has also served as the basis for estimated annual consumption in the contraceptive procurement tables used by USAID to guide procurement and shipments.

Both the summary delivery report and the feedback report are sent to family planning managers at the regions, provinces, and cities, every quarter. In theory, they are invaluable tools for monitoring the performance of each facility, identifying and correcting problems, and providing guidance and supervision. However, it is not clear how regularly and extensively these reports are used by managers to manage the facilities under their supervision.

During the first years of CDLMIS implementation, JSI designed a number of reports that calculate couple-years of protection, determine the peso value of shipped commodities, and flag unusually high or low consumption for types of facilities or individual facilities. As of 1994, none of these reports were being utilized.

In recent years, problems associated with host organization staffing for CDLMIS (described earlier) have adversely affected the utilization of data. In particular, a chronic shortage of people to enter and manage data has caused a severe backlog of forms. This problem was first noted by JSI consultants in 1997, and was reported to persist by the most recent visit in early 2002. As a result of the backlog, summary delivery reports are not available to help determine quantities to supply to provinces and cities. Instead of using current LMIS data to make these decisions, logistics managers are using historical data or their own judgement.

LESSONS LEARNED

- To continue to operate successfully, a computerized LMIS must provide data to support at least one vital logistics decision.
    - In Bangladesh, the computerized LMIS produces information essential to coordinating the activities of various donors. This has created a large and appreciative audience, and helped to sustain demand for LMIS data.

- For most of its existence, the Philippines CDLMIS primarily performed one essential task: providing data to central managers to determine quantities to resupply to provinces and cities. As data entry became backlogged in the late 1990's, CDLMIS could no longer perform this task and, consequently, lost its main usefulness to logistics managers.

- There is always some time delay between collection of LMIS data and the availability of the data in a central, computerized LMIS. Making this data available in time for routine resupply decisions at lower levels may require significant changes to the computerized LMIS.
  - In Bangladesh, the monthly LMIS report produced centrally and then sent to lower levels is not timely enough for resupply decisions made at those levels. Longer-term solutions being explored include (1) sending data electronically to lower-level facilities immediately after processing by the central computerized LMIS; and (2) moving data processing—and thus the computerized LMIS—down to lower-level facilities.
- Successful computerized LMIS include reports and graphs that summarize and present data in ways that are meaningful to analysts, managers, and policymakers.
  - LMIS in both Bangladesh and Kenya generate reports and graphs that highlight overall logistics system performance, such as total quantities dispensed and stockout rates over time. Both systems have gained the notice and support of policymakers and other donors.
  - In Nepal, MOH policymakers and donors have in the past complained about a lack of reports that summarize and analyze data from the computerized LMIS.
  - In the Philippines, the CDLMIS includes reports that were of potential interest to policymakers within DOH—such as a report calculating couple-years of protection—but these reports were never utilized. The lack of demand for CDLMIS data among DOH decision makers appears to be a major factor in the failure of CDLMIS to gain political and financial support within DOH.

Appendix 2

Recommended LMIS Reports and Graphs

STOCK STATUS BY DISTRIBUTION LEVEL

Sample Logistics System
DELIVER Database

Run Date: 23-Jun-03
Run Time: 10:01 AM
Page: 1 of 1

Stock status by distribution level

Report Period: 4th Quarter 1999

Product Facility Level	-#-	Opening Balance	Receipts	Issues	Adjustments	Closing Balance	Months of Stock	% of Total
Amoxycillin 250mg capsule								
1	2	2000	0	1000	0	1000	0.3	70%
3	10	500	1000	300	0	430	0.5	30%
12	500	230	300	0	430	0.8	100%	
Erythromycin 250mg tablet								
1	2	3000	0	400	100	1400	1.2	700%
3	10	800	400	600	0	600	0.3	30%
12	3800	400	1000	-100	0	1.5	100%	
Aspirin 300mg tablet								
1	2	1000	0	500	0	500	0.3	45%
3	10	400	500	300	0	600	0.7	55%
	12	1400	500	800	0	1100	1.0	100%
Vitamin A 200,000IU capsule								
1	2	1500	0	1000	0	500	0.2	36%
3	10	700	1000	800	0	900	0.4	64%
12	2200	1000	1800	0	1400	0.6	100%	

STOCK STATUS BY FACILITY

Sample Logistics System
DELIVER Database

Run Date: 23-Jun-03
Run Time: 10:58 AM
Page: 1 of 1

Stock Status by Facility

Report Period: 4th Quarter 1999

Facility: **Central Warehouse**
Type: MCH Directorate
Supplier: N/A

Minimum months on hand: 6.0
Maximum months on hand: 12.0

Product	Opening Balance	Receipts	Issues	Adjust -ments	Closing Balance	Months of Stock	Average Monthly Cons.	Maximum Stock	Reorder Quantity
Amoxycillin	0	0	0	0	0	0.0	2,267	27,204	28,800
Erythromycin	0	0	0	0	0	0.0	5,667	68,004	72,000
Aspirin	0	0	0	0	0	0.0	5,167	62,004	62,400
Vitamin A	0	0	0	0	0	0.0	2	24	50

STOCK STATUS BY PRODUCT

Sample Logistics System
DELIVER Database

Run Date: 20-Jun-03
Run Time: 2:21 PM
Page:

1 of 1

Stock Status by Product

Report Period: 4th quarter 1999

Product: Amoxycillin 250mg capsule

Facility	Facility Type	Receipts	Issues/ Dispensed	Closing Balance	Months of Stock	Average Monthly Consumption
Central Warehouse	**Central store**	**0**	**1500**	**2800**	**1.6**	**1750**
Dispensing to:						
Arlington District	District store	200	120	500	3.33	150
Ballston District	District store	400	600	450	0.81	550
Fairfax District	District store	300	450	200	0.4	500
Clarendon District	District store	50	220	80	0.44	180
Falls Church District	District store	500	480	300	0.68	440
Happy Babies NGO	NGO	50	35	75	1.66	45

STOCK STATUS ERRORS

Sample Logistics System
DELIVER Database

Run Date: 23-Jun-03
Run Time: 10:34 AM
Page: 1 of 1

Stock Status Errors
Report Period: 4th Quarter, 1999

Product: **Amoxycillin 250mg capsule**

Facility		Facility Type	Receipts	Issues
Central Warehouse 100		MCH Directorate	0	**0**
Dispensing to:	Arlington District	District Store	**230**	300
	Fairfax District	District Store	**100**	150
		Total	330	450

DATA ENTRY ERRORS

Sample Logistics System
DELIVER Database

Run Date: 20-Jun-03
Run Time: 2:21 PM
Page: 1 of 1

Data Entry Errors

Report Period: 4th quarter 1999

Facility: **Arlington District**
Facility type: District warehouse
Supplying facility: Central Warehouse 100

Product	Opening Balance	Receipts	Issues	Adjust -ments	Adjust. Type	Closing Balance	Average Monthly Cons.	Quantity Required
Amoxycillin	500	230	300	0	-	430	2267	6371
Erythromycin	0	0	0	0	-	0	0	0
Aspirin	0	0	0	0	-	0	0	0
Vitamin A	0	0	0	0	-	0	0	0

NON-REPORTING FACILITIES

Sample Logistics System
DELIVER Database

Run Date: 23-Jun-03
Run Time: 10:32 AM
Page: 1 of 1

Non-Reporting Facilities

Report Period: 4th quarter 1999

Facility	Type	Location	
Central Warehouse			
Washington DC Distribution Center	MCH Directorate	Washington	
Richmond Distribution Center	Health Directorate	Richmond	
Winchester Distribution Center	Health Directorate	Winchester	
Arlington Wilson Street	Health Center		
Arlington 4	Health Center		
SDP -Test Arlington	Health Center	Arlington	
		Non-reporting facilities:	6
		Non-reporting percentage:	75%
Charlottesville District Store			
Charlottesville 1 (UVA)	JAFPP Clinic	Charlottesville	
		Non-reporting facilities:	1
		Non-reporting percentage:	100%
Richmond Distribution Center			
Richmond 1	Health Center	Richmond	
Cold Harbor	Health Center	Cold Harbor	
Petersburg	Health Center	Petersburg	
		Non-reporting facilities:	3
		Non-reporting percentage:	100%
Washington DC Distribution Center			
Test SDP 1	Health Center	Washington	
SDP 3-Bethesda	MOH Hospital	Bethesda	
		Non-reporting facilities:	2
		Non-reporting percentage:	100%
Winchester Distribution Center			
Cedar Creek Center	Health Center	Cedar Creek	
Fisher Hill	Health Center	Winchester	
		Non-reporting facilities:	2
		Non-reporting percentage:	100%

Total non-reporting facilities: 14

Total non-reporting percentage: 82%

STOCK IMBALANCES

Sample Logistics System
DELIVER Database

Run Date: 23-Jun-03
Run Time: 10:32 AM
Page: 2 of 2

Stock Imbalances

Report Period: 4th Quarter, 1999

Supplier: **Central Warehouse**

Facility	Product	Status	Closing Balance	Avg. Monthly Consumption	Months of Stock	Quantity Needed
Arlington Test 2	Amoxycillin	Stocked Out	0	5,667	0.0	17,001
	Erythromycin	Stocked Out	0	2	0.0	6
	Paracetamol	Below Minimum	430	2,267	0.2	6,371

ADJUSTMENTS SUMMARY

Sample Logistics System
DELIVER Database

Run Date: 23-Jun-03
Run Time: 10:36 AM
Page: 1 of 1

Adjustments Summary

Report Period: Year 2003

Product / Facility Name	Facility Type	Supplier	Adjustment Type	Adjusted
Amoxycillin 250mg capsule				
Test SDP 1		Washington DC Distribution	DE Credit	1,087
			Total DE Credit	*1,087*
Washington DC Distribution		Central Warehouse 100	DE Debit	-500
			Total DE Debit	*-500*
Central Warehouse 100			Expired	-10,000
			Total Expired	*-10,000*
		Total Adjustments, Amoxycillin:		**1,077**
Erythromycin 250mg tablet				
Central Warehouse 100			DE Debit	300
			Total DE Debit	*300*
		Total Adjustments, Erythromycin:		**300**
Aspirin 300mg tablet				
Arlington Test 2		Central Warehouse 100	Damaged	-1
			Total Damaged	*-1*
Cold Harbor		Richmond Dist Center	DE Debit	-2
			Total DE Debit	*-2*
		Total Adjustments, Aspirin:		**-3**

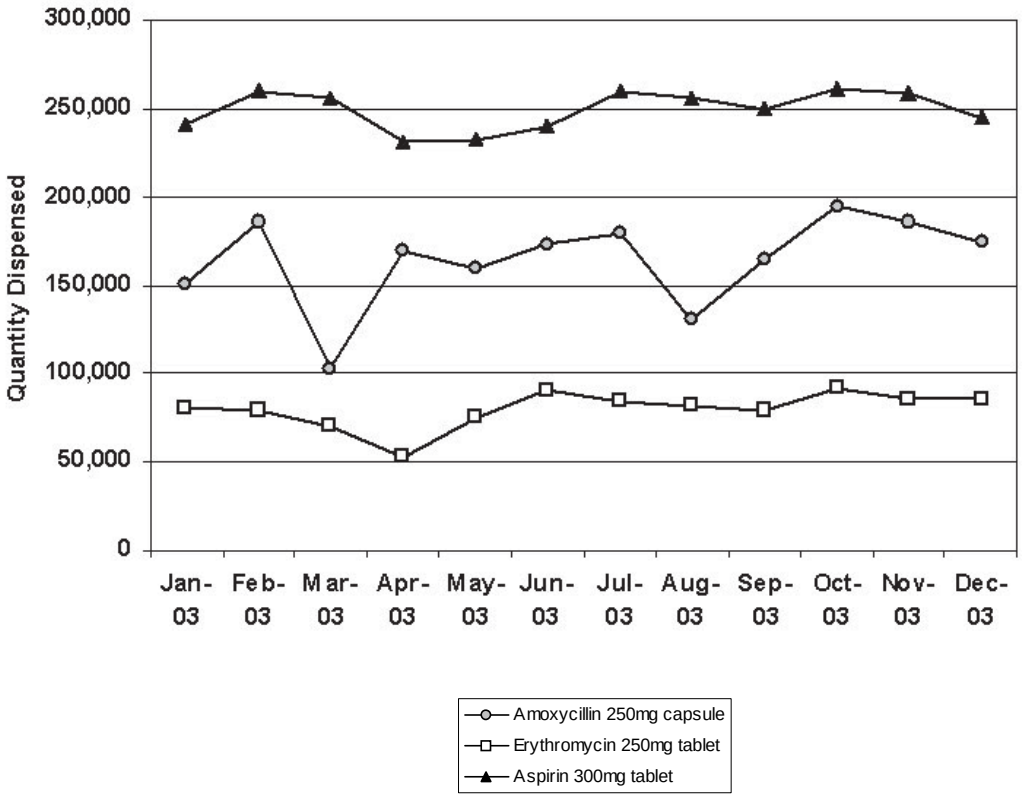
DISPENSED TO USERS

Sample Logistics System
DELIVER Database

Run Date: 23-Jun-03
Run Time: 10:36 AM
Page: 1 of 1

Dispensed to Users

Report Period: Year 2003



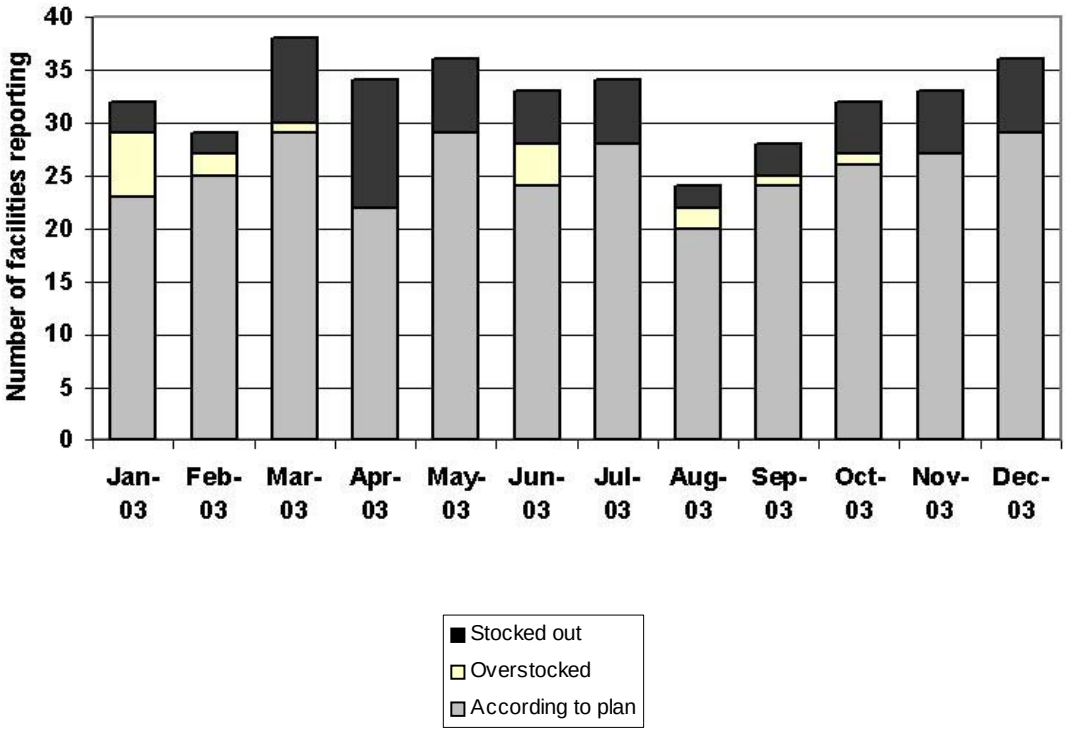
STOCK STATUS

Sample Logistics System
DELIVER Database

Run Date: 23-Jun-03
Run Time: 10:36 AM
Page: 1 of 1

Stock Status

Report Period: Year 2003



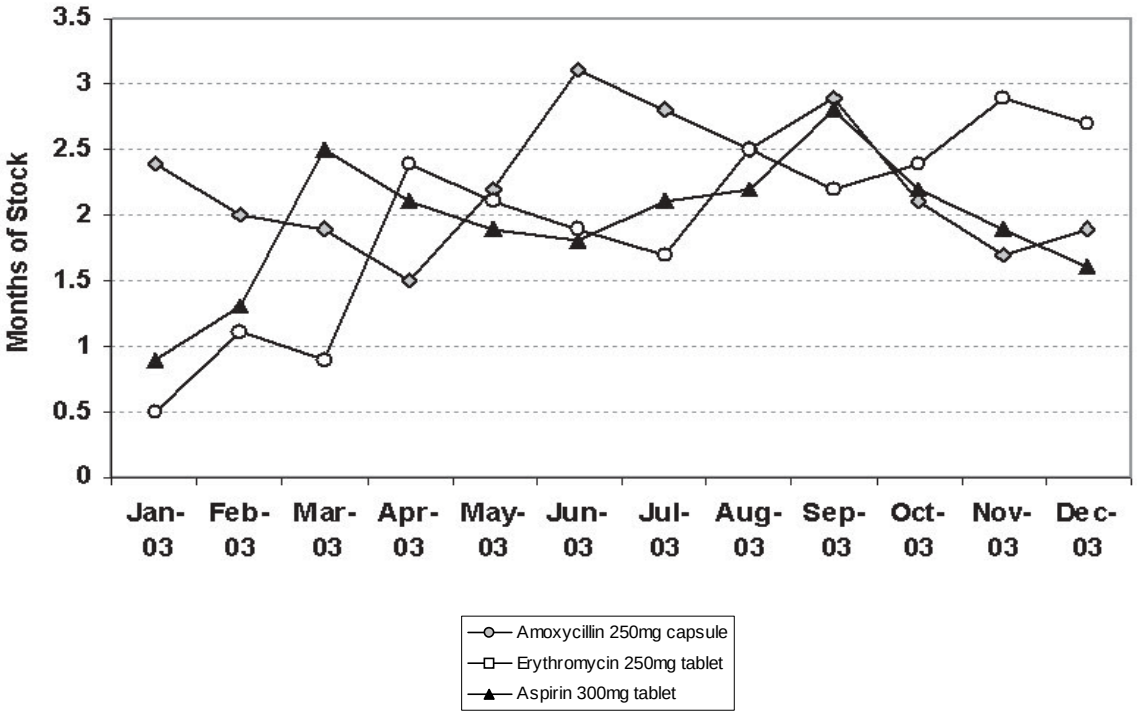
AVERAGE STOCK LEVELS

Sample Logistics System
DELIVER Database

Run Date: 23-Jun-03
Run Time: 10:36 AM
Page: 1 of 1

Average Stock Levels

Report Period: Year 2003



Appendix 3

LMIS Development Process Documents

REQUIREMENTS ASSESSMENT OUTLINE

Introduction

Problem statement

[Briefly describe the current distribution system and especially the impetus for seeking to implement a new computerized LMIS or change an existing computerized LMIS. What is the root problem driving this activity? This section should be short—no more than three to four paragraphs.]

Purpose

[Explain how the proposed new or changed system will address the root problem described above. This section should also be brief. It is often helpful to describe the new system functions in a bulleted list.]

Scope

[Use this section to state what the system will and will *not* do. A common misapprehension held by project sponsors and end users at the start of a development project is that the new or changed system will solve all their problems. This section provides an opportunity to clarify from the beginning that the proposed system will solve some but not all of their problems.]

Assumptions

[Describe any tasks that the client needs to complete before the new system can be implemented successfully. If there are no such tasks, leave this section out.]

System Overview

Procedures

[New computerized systems almost always require new procedures. Use this section to outline any major changes in procedures that will be required by the new system. In some cases, this section may be short; in others, it may be longer, depending on the extent and scope of procedural changes required.]

Users

[List the types of users of the system and how they will interact with the new system. A bulleted list is usually sufficient for this section.]

Technology

[This section provides an opportunity to describe and evaluate various technology options. For truly new systems, the two options are usually "buy vs. build." For enhancements to existing systems, this section can be used to assess various approaches or software tools to support the enhancements.]

System Functions

[This section should include one and only one thing: use cases.]

Other Requirements

Security

[Describe the security model for the new system. A simple way to do this is to include a bulleted list of user types, along with system functions that they can access.]

Training

[Outline the training that will be needed to make the system operational. Like the security section, this section can contain a bulleted list of user types, plus the training that they will receive prior to system deployment.]

Documentation

[Briefly describe the written materials that will be produced, including any technical manuals.]

Technical support

[Explain how the system will be supported: what do users do when they encounter problems, or the system breaks down entirely? This section should briefly describe who provides primary technical support, and who s/he can turn to for problems that cannot be resolved on-site.]

Development and Implementation Process

[Provide a high-level timetable for developing and implementing the system. Like a workplan, this section can include a list of tasks and the expected completion date for each task.]

Illustrative System Outputs

[Design mock reports and present them here. As much as possible, these reports should support tasks described in previous sections and include actual data.]

REQUIREMENTS ASSESSMENT EXAMPLE
Warehouse Management System (WMS)

1 Introduction
- 1.1 Problem statement
- 1.2 Scope
- 1.3 Purpose

2 System Overview
- 2.1 Procedures
- 2.2 Users
- 3.3 Technology

3 System Functions
- 3.1 Record a purchase order
- 3.2 Receive a purchase order
- 3.3 Receive expired items from field sites
- 3.4 Remove expired or damaged items
- 3.5 Issue items—paper requisitions
- 3.6 Determine quantities to order
- 3.7 Manage the list of supplies
- 3.8 Conduct physical inventories
- 3.9 Move items between storage locations
- 3.10 Issue items—partial orders
- 3.11 Request items—electronic requisitions
- 3.12 Issue items—electronic requisitions

4 Other Requirements
- 4.1 Security
- 4.2 Training
- 4.3 Documentation
- 4.4 Technical support

5 Development and Implementation Process

6 Illustrative System Outputs
- 6.1 Stock movement report
- 6.2 Physical inventory report
- 6.3 Pick list
- 6.4 Consumption report—by item
- 6.5 Consumption report—by customer (facility)

7 Process flow diagrams
- 7.1 Requisition and issue items
- 7.2 Record purchase order
- 7.3 Receive purchase order
- 7.4 Return expired items
- 7.5 Conduct physical inventory
- 7.6 Reorder items

1 Introduction

1.1 Problem statement

A reliable supply of laboratory reagents, pharmaceuticals, and other items is critical to Rosslyn's activities in the areas of HIV testing, research, surveillance, prevention, and treatment. In recent months, supplies have not been reliable; in fact, stockouts of essential items have disrupted project activities. At the same time, overstocks of some items have led to wastage of expired products.

There are two primary causes for the stock imbalances. First, the project does not have a standard system for tracking stock movement and use. As a result, project staff are unable to routinely monitor stock levels and take action to correct stock imbalances (either by placing orders for resupply or by redistributing stock about to expire). Second, the project's procurement process involves a number of steps across several organizations; although each step is usually performed efficiently, the sheer number of steps required means that the lead time for some orders is six months or more. These delays can lead to stockouts even when orders are placed several months in advance.

1.2 Scope

The system proposed in this document aims to address the first of the root problems mentioned above by providing a standard way to track stock movement and use.

Tracking the procurement process—from internal project request to submission of a purchase order to the vendor—is out of the scope of this system. While closely related to stock management, procurement is essentially a separate process; developing a system to track this process would require a separate set of activities that could take place either in parallel with the implementation of the stock management system or afterwards. (The system described here will, however, enable stock managers to record a purchase order after it has been submitted to a vendor for delivery.) Focusing first on implementing a stock management system is likely to yield the greatest benefit to Rosslyn: effective management of current supplies—which this system will facilitate—is the foundation of all logistics activities, including procurement. After this system is in place, Rosslyn can then use the information it generates to better manage procurement activities.

Also outside the scope of this system is stock management at field sites. After this system is successfully implemented at project headquarters, it could be extended in the future to include stock management at field sites.

1.3 Purpose

The purpose of the system is to track the movement of laboratory supplies and pharmaceuticals into and out of storage locations at the project headquarters and offsite. Information recorded in the system will enable Rosslyn to make timely stock management decisions. In particular, the system will provide data for Rosslyn staff to perform the following essential tasks:

- Estimate how long current supplies will last based on average consumption.
- Calculate quantities of items to order based on average consumption and current supplies in stock.

- Determine when to replenish on-site storage locations with supplies from backup storage locations.
- Identify products that are about to expire in order to redistribute or otherwise dispose of them.

2 System Overview

2.1 Procedures

Implementation of the computerized system depends on the establishment of new, standard stock management procedures. To facilitate the short-term success as well as the longer-term sustainability of the stock management system at Rosslyn and any successor project, the procedures proposed in this document build on existing procedures, and rely largely on technology familiar to project staff.

The most fundamental of these new procedures are (1) limiting access to storerooms to two designated staff per section; and (2) requesting and issuing supplies via a requisition form. These two procedures are already in place for some supplies (pharmaceuticals and non-refrigerated clinical lab supplies).

The new procedures are described in detail in section 3:

- Record a purchase order.
- Receive a purchase order.
- Receive expired items from field sites.
- Remove expired or damaged items.
- Issue items—paper requisitions.
- Determine quantities to order.
- Manage the list of supplies.
- Conduct physical inventories.
- Move items between storage locations.
- Issue items—partial orders.
- Request items—electronic requisitions.
- Issue items—electronic requisitions.

2.2 Users

There are four primary groups of users of the system:

Stock managers for the clinical lab, virology, and the administrative section will be the most active users of the system. They will receive and issue all supplies, and record the transactions in the system. They will use reports generated by the system to calculate quantities to order based on past consumption. They will also monitor stock levels and alert section chiefs to impending problems or issues (such as critically low supplies or supplies about to expire).

Section chiefs will use reports generated by the system to monitor consumption of laboratory supplies and pharmaceuticals.

Administrative section staff will use reports generated by the system to verify receipt of purchase orders, and may also use reports to validate purchase order requests from other sections based on current supplies and rates of consumption.

Technicians and field site staff will submit a requisition form to stock managers to request supplies. In a later implementation phase, technicians at project headquarters will submit requisition forms electronically through the system.

2.3 Technology

The computerized system will be based on a U.S.-made stock management software package (Smart WMS) that can be configured to operate in multiple languages. The advantage of using a package, rather than developing a custom solution from scratch, is that it gives Rosslyn access to a range of external resources for customization and technical support. Using a package solution also gives Rosslyn the option to upgrade to new versions of the software as they are released.

Another advantage of Smart WMS is because source code is provided, Rosslyn can independently modify the package to suit its needs in the future. The project could enlist internal resources (staff from the data section) to modify and support the software, or it could hire a local technology company to do the work. The significant disadvantage of this approach is that it precludes technical support from the manufacturer, and it makes upgrading to new releases more difficult. But, it does enable Rosslyn to find independent resources to maintain the system if existing resources (such as technical support or customization services provided from outside Cote d'Ivoire) become unavailable or not workable in the future.

Smart WMS will be set up on multiple (five or more) existing desktop computers connected to a database server on Rosslyn's computer network. Stock managers will use these computers to manually enter data on stock movement into the system; other users will use the computers to access reports generated by the system. This architecture will utilize Rosslyn's current technology infrastructure making it relatively easy to operate and maintain.

While Smart WMS can be operated with scanners for reading barcode labels on items—thus making data entry easier and more accurate—the costs and risks of introducing scanners at Rosslyn outweigh the potential benefits, at least at this time. An implementation based on barcoded items would require equipment that is relatively difficult [to obtain?] and costly to maintain. In addition to scanners, the project would need special bar code printers and labels, as well as mechanisms for transferring inventory data from scanners to the host database (via direct cable connections or radio signals). There would also be extra labor costs associated with printing and applying labels to all incoming shipments.

The procedures proposed in this document—designating a limited number of users (stock managers) to enter data—can be an effective means to ensure the accuracy of inventory data, without the risks and costs associated with special equipment. If the stock management needs of Rosslyn change in the future, barcoding can be implemented using the built-in features of Smart WMS.

3 System Functions

3.1 Record a Purchase Order

1. After the administrative section has sent a purchase order to the vendor for delivery, a copy of the purchase order is given to the stock manager of the section ordering the supplies.
2. The stock manager records a new receiving order in the system:
   - purchase order number
   - purchase order date
   - vendor
   - expected delivery date
   - item number
   - quantity.

3.2 Receive a Purchase Order

1. When items are received and put away, their location within the storeroom is noted on the receiving documents.
2. Using the receiving documents, the stock manager then finds and updates the purchase order record in the system:
   - storeroom
   - location within storeroom
   - quantity received
   - expiration date
   - {optional: lot number}.

3.3 Receive Expired Items from Field Sites

1. When field sites return expired items, the stock manager records the transaction in the system:
   - field site returning the items
   - item number
   - quantity
   - expiration date
   - {optional: lot number}.
2. The system generates a return issue voucher for the field site representative to sign.
3. The stock manager puts the returned items in a quarantine location designated for expired or damaged items.

3.4 Remove Expired or Damaged Items

1. When items are removed from current inventory due to expiry or damage, the stock manager records the transaction in the system:
   - original storeroom and location
   - item number
   - expiration date
   - quantity removed
   - reason for removal (expiry or damage).
2. The system moves the items out of current inventory.

3.5 Issue Items—Paper Requisitions

1. The stock manager receives a requisition form (signed by the section chief if the request originates from a field site), and records it in the system as a new pick order:
   - requester (customer)
   - date of request
   - date needed by
   - item number
   - quantity.
2. The system generates a pick list showing locations of the items to be picked, based on first-to-expirre, first-out (FEFO): items that will expire soonest are selected first.
3. The stock manager picks the items from the storeroom, and notes the actual picked quantities on the pick list.
4. On delivery, the requester signs the pick list to confirm receipt of the items.
5. The stock manager then updates the pick order in the system by recording the quantity of each item picked and delivered.

3.6 Determine Quantities to Order

1. On a regular basis, the stock manager prints a report on stock movement (see sample report in section 6.1) and reviews quantities on hand and quantities to order.
2. The stock manager compiles a list of items to order and submits the list to the administrative section.

3.7 Manage the List of Supplies

1. When new items are added to the list of supplies or there are changes to the characteristics of an item, the stock manager of each section adds an item record or updates an existing record with the following:
   - item number
   - item description
   - category
   - cycle count period (days between physical counts) [later implementation]
   - replenishment threshold (quantity which triggers a pick list to move items from backup storage) [later implementation]
   - primary location [optional].

3.8 Conduct Physical Inventories

1. On a regular basis (weekly or monthly), the stock manager prints a physical inventory report listing items due to be inventoried (based on cycle period established for that item), including their locations and expiration dates.
2. The stock manager does a physical count, and writes down the quantities on the inventory report, with any discrepancies in location or expiration date.
3. The manager then records the actual quantities of each item in the system; notes the reason for any discrepancy, if known; and updates the expiration date if necessary.

3.9 Move Items between Storage Locations

1. On a regular basis (daily or weekly), the stock manager of each section prints a pick list of items to be moved from the backup storeroom(s) to the primary storerooms at project headquarters.
2. After the items have been moved from the backup to the primary storerooms, the stock manager records the transaction in the system:
   - originating storeroom and location
   - destination storeroom and location
   - item number
   - quantity
   - expiration date.

3.10 Issue Items—Partial Orders

This feature will enable the stock manager to track orders that have been filled partially, and to issue items for part of an order. (Specifics to be defined during phase 1 implementation.)

3.11 Request Items—Electronic Requisitions

1. To request an item, headquarters staff enter a new pick order in the system (previously done by the stock manager):
   - requester (customer)
   - date needed by
   - item number
   - quantity.
2. The system records the current date as the date of request.

3.12 Issue Items—Electronic Requisitions

1. At least once a day, the stock manager prints a pick list of new requisitions placed by headquarters staff.
2. The stock manager then follows the process for issuing items (paper requisitions).

4 Other Requirements

4.2 Security

All users must have a valid username and password to access the system. (The system may use a valid network login to authenticate users.)

Each user will be assigned a set of privileges to access certain parts of the system:

- System administrator: system setup, security, and language features.
- Stock managers: inventory features.
- Other staff (primarily section chiefs): reporting features.

4.2 Training

There will be four types of training sessions tailored to each group of users of the system:

- System administrator: system configuration, maintenance, and technical support.
- Stock managers: receiving, issuing, and moving stock; monitoring stock movement and inventory levels.
- Other staff (primarily section chiefs): accessing and interpreting reports on stock movement and inventory levels.
- Technicians (and field staff): submitting requisition forms.

4.3 Documentation

In addition to the standard Smart WMS documentation (user's guide), the following materials will be developed to document the customized implementation of Smart WMS at Rosslyn:

- Procedures manual (user's guide) for managing Rosslyn stock using Smart WMS.
- Technical manual for Rosslyn data section staff, describing configuration and maintenance of database server, workstations, and customized application.

4.4 Technical Support

The system administrator will provide on-site technical support to users of the system in resolving technical problems. For problems that cannot be resolved on site, technical support will be provided by Smart WMS offices based either in the U.S. or in France. Options for technical support are currently under discussion with Smart WMS representatives. Specific mechanisms and procedures for technical support will be defined before and during the first phase implementation.

5 Development and Implementation Process

The features listed in section 3 will be implemented in two or more phases. The following features will be implemented during the first phase:

3.1 Record a purchase order.
3.2 Receive a purchase order.
3.3 Receive expired items from field sites.
3.4 Remove expired or damaged items.
3.5 Issue items –paper requisitions.
3.6 Determine quantities to order.
3.7 Manage the list of supplies.

The following features will be implemented during the second phase:

3.8 Conduct physical inventories.
3.9 Move items between storage locations.
3.10 Issue items–partial orders.

The last two features, related to electronic requisitions, may be implemented during the second phase, or may be deferred for a later phase:

3.11 Request items–electronic requisitions.
3.12 Issue items–electronic requisitions.

6 Illustrative System Outputs

6.1 Stock Movement Report

Report selection criteria:

- Product category
- Site
- Date range.

CATEGORY: **Clinical laboratory** **FROM:** **January 1, 2001**
SITE: **{All}** **TO:** **March 31, 2001**

Product	Opening Balance	Receipts	Issues	Adjust -ments	Ending Balance	Average Monthly Usage	Months of stock	Qty. to Order*
Pipet 10 ml	5	10	12	0	3	4	.75	21
Pipet 25 ml	22	0	15	0	7	5	1.4	23
Cyrovial 4ml	30	0	17	-3	10	5.67	1.76	26

*Quantity to order = Average monthly usage x 6 (maximum months of stock) minus ending balance

6.2 Physical Inventory Report

Report selection critieria:

- Site

SITE: Clinical lab stores **DATE: February 6, 2002**

Item	Date last counted	Projected stock on hand	Actual stock on hand	Reason for discrepancy
Pipet 10 ml	12/30/01	3		
Pipet 25 ml	12/30/01	7		
Cyrovial 4ml	12/30/01	10		

6.3 Pick List

Report selection critieria:

- Order number
- Order date
- Due date.

ORDER #: **12345**
CUSTOMER: **Clinique de Confiance**
ORDER DATE: **1/2/2002**
DUE DATE: **2/28/2002**

Item #	Description		Ordered Qty	Previous Issued Qty	Balance
23456	**Pipet 10 ml**		**500**	**100**	**400**
	Location	UOM	On hand Qty	Exp. Date	Qty to Pick
	B22	EACH	200	1/31/2003	200
	C34	EACH	500	3/31/2003	200
45678	**Cyrovial 4ml**	**200**	**0**	**200**	
	Location	UOM	On hand Qty	Exp. Date	Qty to Pick
	B12	EACH	50	1/31/2003	50
	C7	EACH	500	3/31/2003	150
67984	**Reactif**	**10**	**0**	**10**	
	Location	UOM	On hand Qty	Exp. Date	Qty to Pick
	COLD	EACH	23	6/30/2002	10

Received by _____

Date _____

6.4 Consumption Report—By Item

Report selection criteria:

- Item
- Product category
- Site
- Date range.

CATEGORY: **Clinical laboratory** **FROM: January 1, 2001**
SITE: **{All}** **TO: March 31, 2001**

Item #	Description	Customer	Issue Date	Issued Qty
23456	Pipet	Clinique de Confiance	1/30/01	100
		Maternity Clinic #32	2/3/01	200
		TB clinic #45	3/15/01	200
		Maternity Clinic #56	3/22/01	300
			Total pipet issued	**800**
33456	Cyrovial	Clinique de Confiance	1/30/01	50
		TB clinic #78	2/4/01	100
			Total cyrovial issued	**150**

6.5 Consumption Report—By Customer (Facility)

Report selection criteria:

- Customer
- Product category
- Date range.

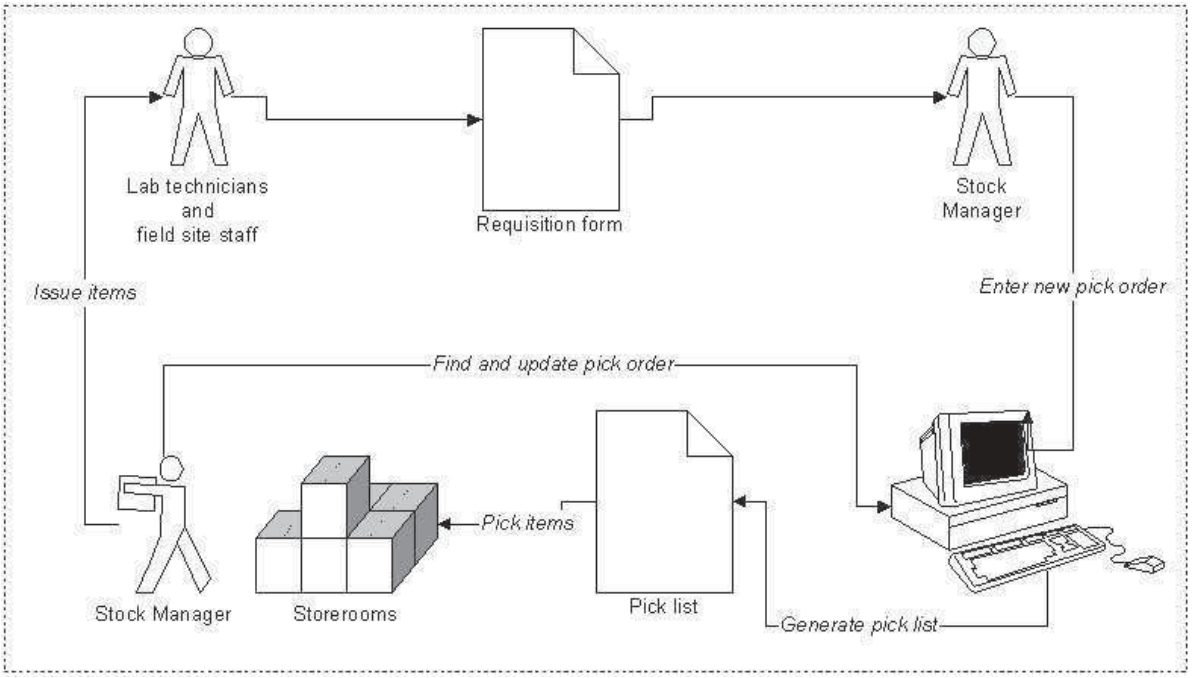
CUSTOMER: Clinique de Confiance

FROM: January 1, 2001
TO: March 31, 2001

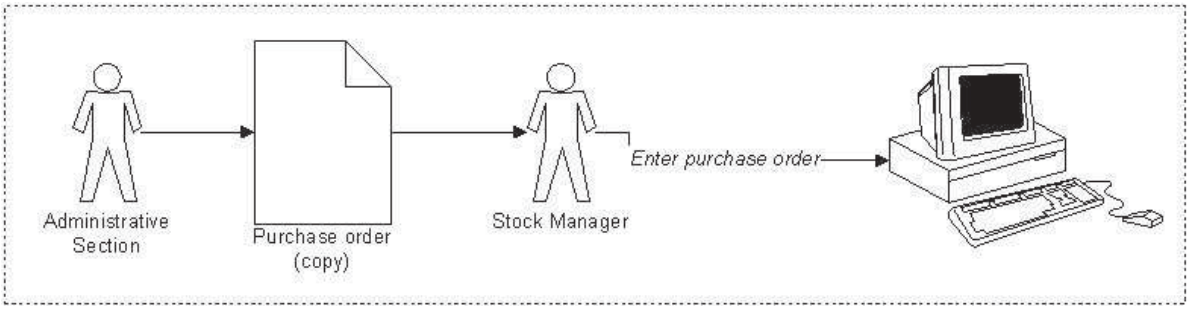
Category	Item #	Description	Issued Qty
Clinical Labortory			
	23456	Pipet	100
	33456	Cyrovial	50
Pharmaceuticals			
	45678	Paracetamol	100

7 Process Flow Diagrams

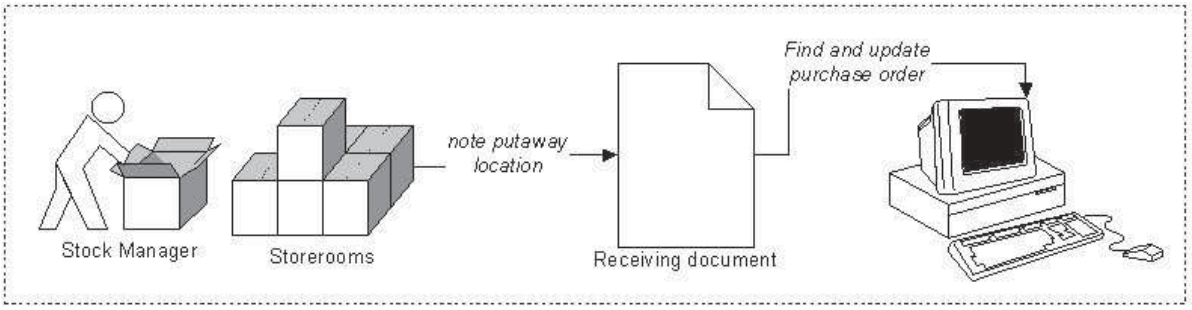
7.1 Requisition and Issue Items



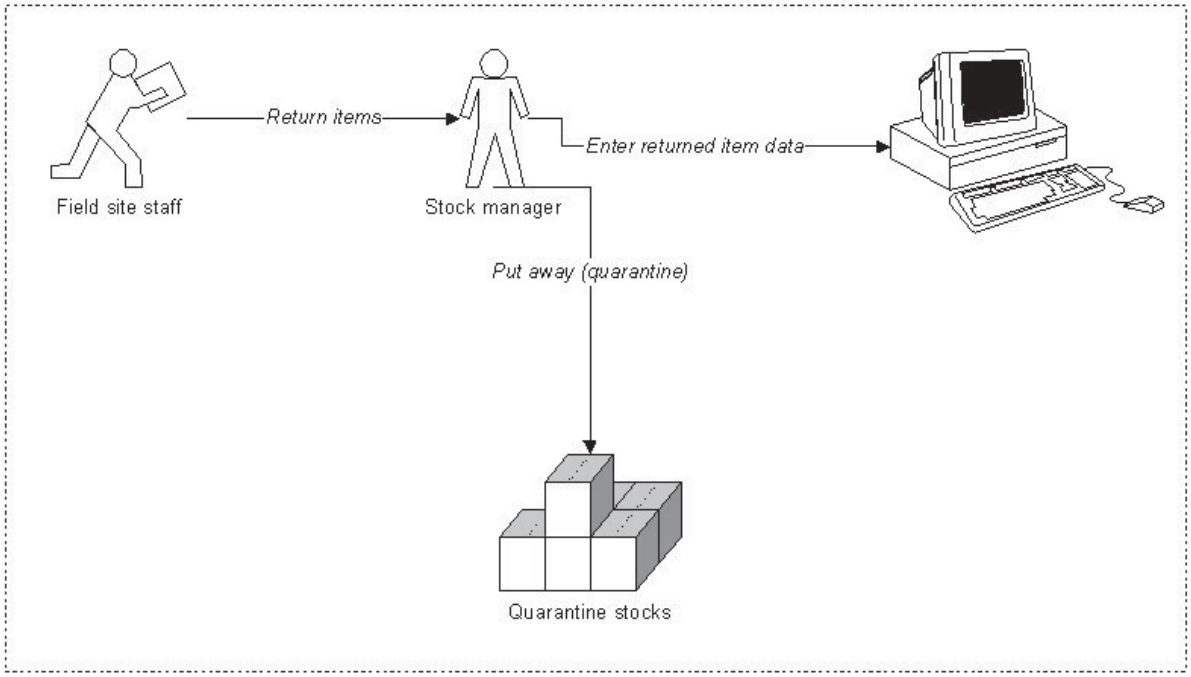
7.2 Record Purchase Order



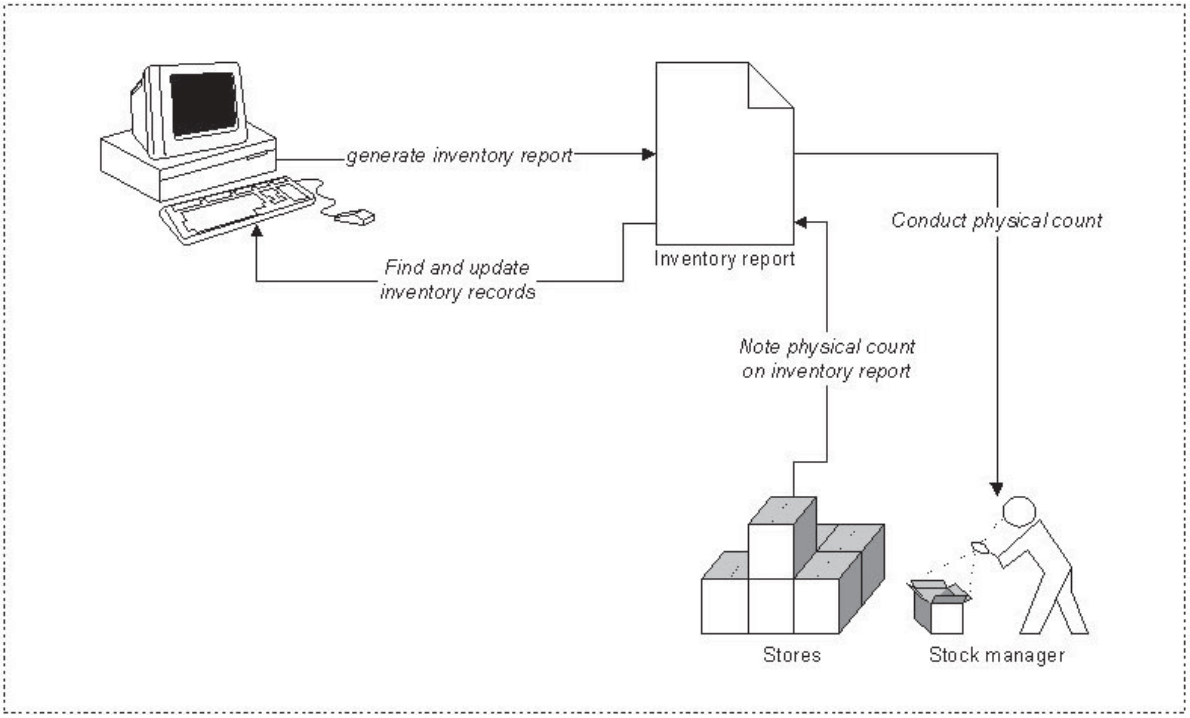
7.3 Receive Purchase Order



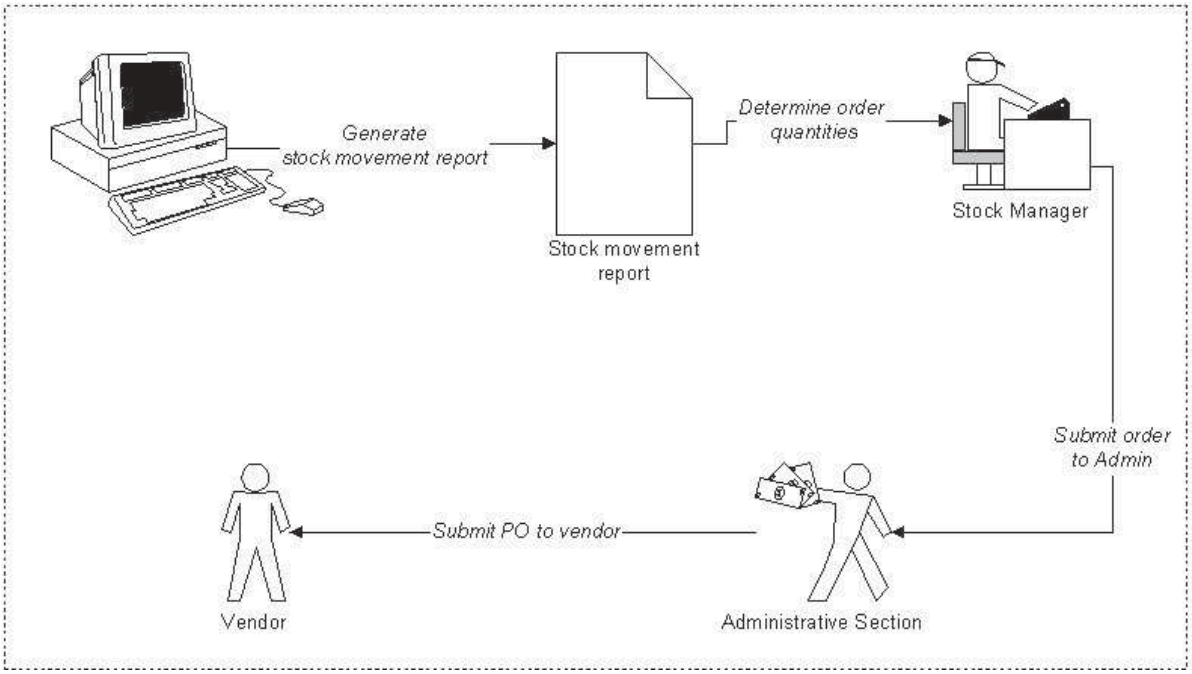
7.4 Return Expired Items



7.5 Conduct Physical Inventory



7.6 Reorder Items



USE CASE

Update a Customer Order

Summary

After an order is shipped, warehouse staff confirms the shipped quantities on the order, and returns the order to the WMS operator. The WMS operator then updates the order in the system.

Main course of events

1. User selects an order number.
2. System finds and displays order details.
3. User selects a pick location, and enters the actual quantity picked for each item in the invoice.
4. System records the quantity as issued and generates an invoice.

Extensions

2a. Order shipped (quantities already issued):

.1 System displays order details in disabled fields. Steps 3 and 4 of the main course of events are skipped.

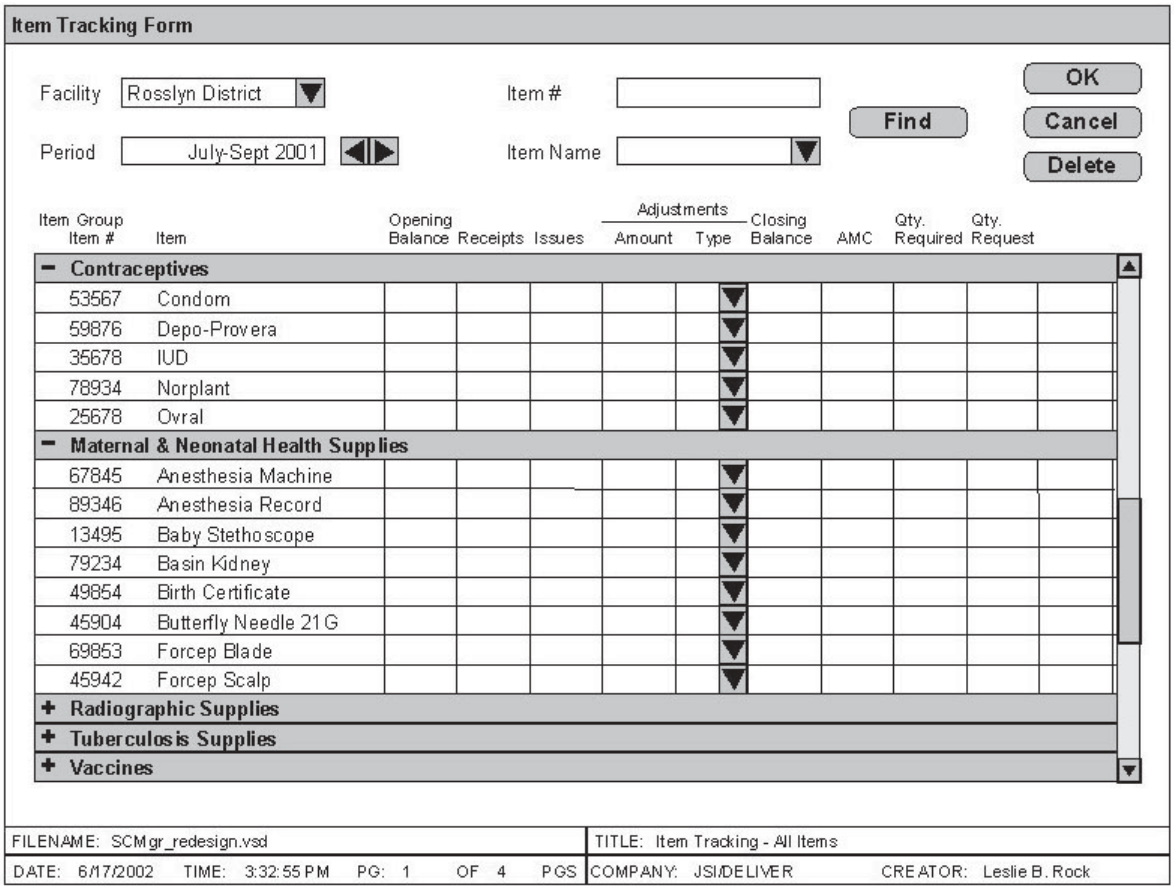
3a. Partial order to be closed (quantities shipped are less than quantities requested, but the order will not remain open):

.1 User closes the order.

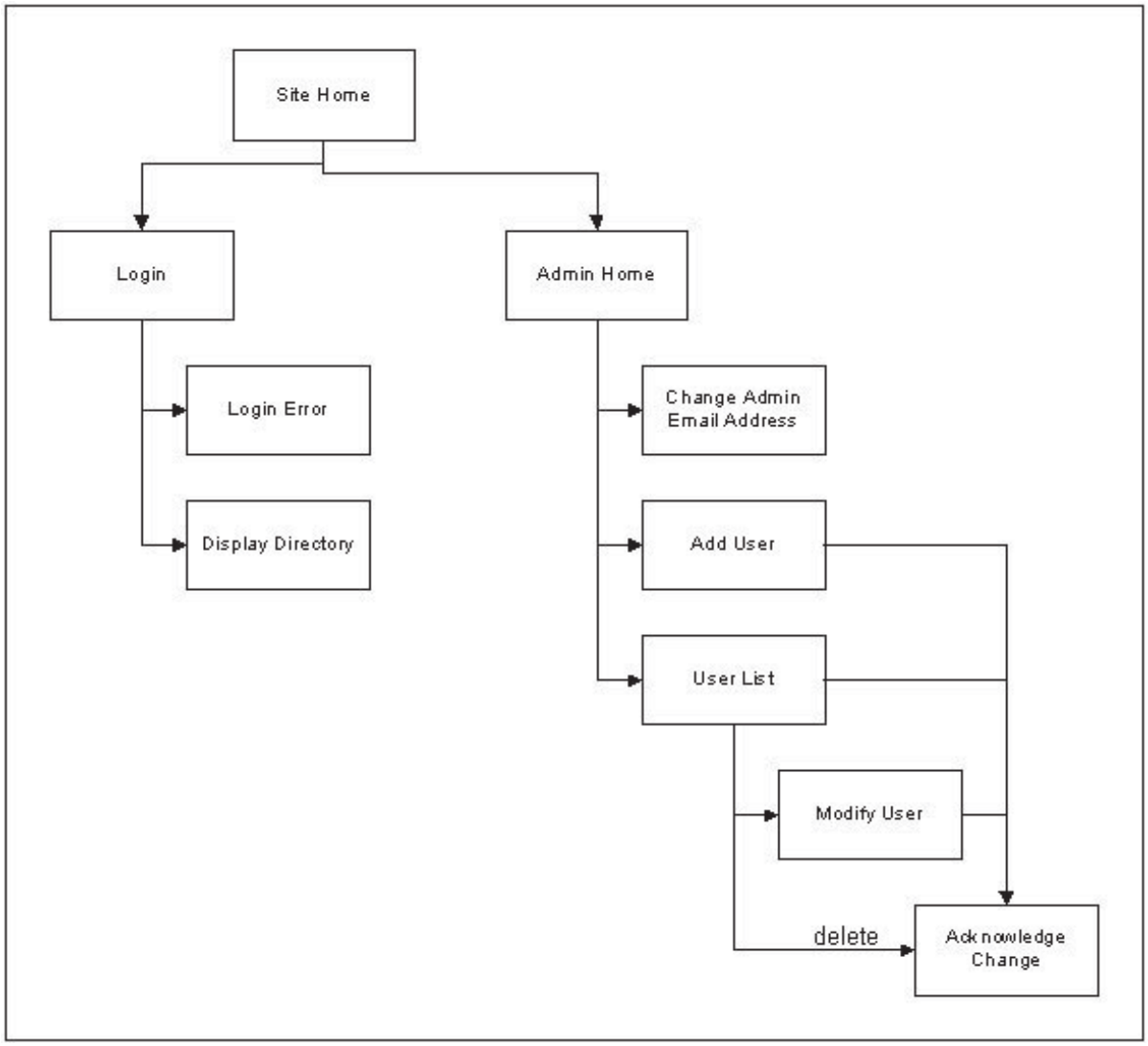
3b. Quantity updated by user is greater than quantity on hand:

.1 System prompts user to edit the quantity entered.

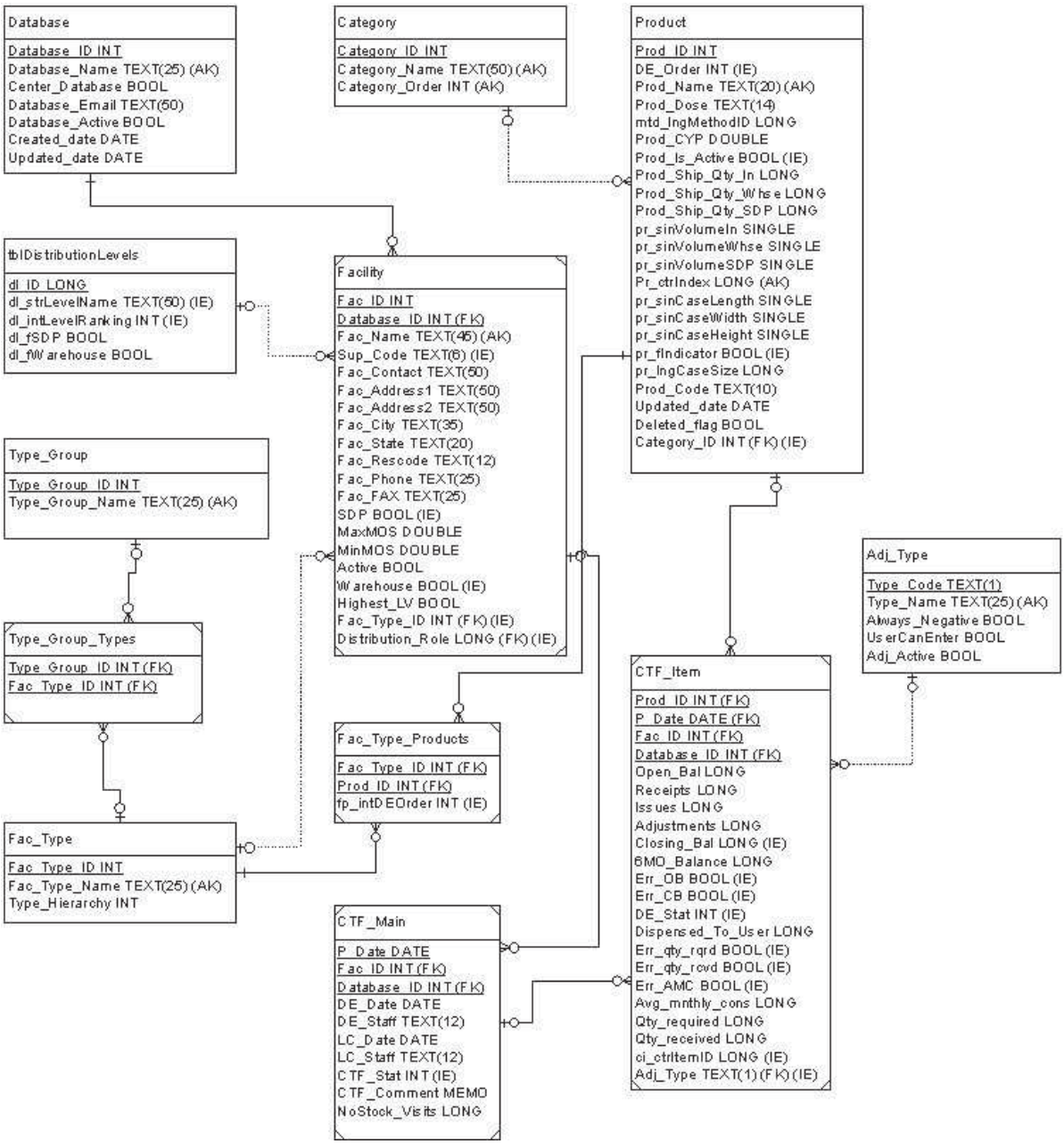
USER INTERFACE PROTOTYPE



SITE MAP


Site Home
Login
Admin Home
Login Error
Display Directory
Change Admin Email Address
Add User
User List
Modify User
delete
Acknowledge Change

DATABASE ENTITY-RELATIONSHIP DIAGRAM



ENHANCEMENT SPECIFICATION

Date: 21-Mar-2003

Enhancement Specification Form

Module Issue Warehouse Memo

Brief description of enhancement _____

Developer _____

User contact Lois/Lesley

Target release date _____

Specifications:

1. Attach prototype or description of user interface (form or report).
2. Describe the functionality/purpose of each element.

Include—

a. *Actions* to be performed by the user and system.
b. *Procedures* or logic for each action.
c. Specific *tests* for each element and the criteria for passing.

Action/Event	Procedure/Functionality	Test
User enters an order number, hits get record icon.	Shipments for order are retrieved into table (existing functionality).	Verify.
User selects shipment where status is "P" or "0."	Bottom box is activated (existing functionality).	Verify.
User checks new field labeled "Specify a Lot:"	Box is checked.	New field located underneath "Reallocate Stock" field.
User may deselect "Draft:" box—no restriction.	Existing functionality.	Verify.
User clicks on "Issue WH Memo" button.	If the shipment is a future shipment, message appears; if shipment has already shipped, message appears; etc. (existing functionality). - If shipment is not status "P," message appears " You may only specify a lot for shipments with status s_name where s_code = "P"* with button <OK>.	Test all scenarios.

TEST CASE

Tickler Report: Shipments Awaiting Shipping Documents

Module summary

This report alerts users to shipments that have shipped but the documents have not been received from the freight forwarder.

Menu Selection

Main Menu > Reports Menu > Tickler Reports

Modules

Module	File Name
Tickler Reports Form	TICREP01.fmb
Tickler Report	TICREP01

Interface Test

Sl	Tests
1.	Form selection screen follows NEWVERN reports form standards.
2.	Report follows NEWVERN report standards.

Business Rules and Related Tests

Sl	Business Rules	Test plans
Report		
1.	Report includes all non-deleted shipments, status "shipped" except those to a warehouse where "Receiving Report Sent" field (shipment table) is null.	- Verify with system query - Compare to Examine a Shipment

Verify using NEWVERN data

Query to check underlying data–Shipment Awaiting Shipping Documents Report

TEST CASE (CONT.)

The following query can be run to check the data:

SELECT NEWVERN_SHIPMENT.O_NUM, NEWVERN_SHIPMENT.S_NUM, NEWVERN_CUSTOMER_ORDER.CU_CODE, NEWVERN_CUSTOMER_ORDER.R_CODE, NEWVERN_SHIPMENT.S_DEL_FLAG, NEWVERN_SHIPMENT.S_TWOWAY_DT

FROM NEWVERN_SHIPMENT INNER JOIN NEWVERN_CUSTOMER_ORDER ON NEWVERN_SHIPMENT.O_NUM = NEWVERN_CUSTOMER_ORDER.O_NUM

WHERE (((NEWVERN_SHIPMENT.S_DEL_FLAG)<>"Y") AND ((NEWVERN_SHIPMENT.S_TWOWAY_DT) Is Not Null) AND ((Right([NEWVERN_CUSTOMER_ORDER]![R_CODE],3))<>"WHS"));

Use SQL query results to verify report data:

SI		
1.	Run query with criteria— Tables and fields: Shipment: (o_num, s_num, s_del_flag, s_twoway_dt	Query runs.
2.	Compare query results with report.	Verify results.

Scenario-Based Test

SI		
1.	Generate receiving report for shipment on report.	- Verify shipment no longer appears on the report. - Verify correct updates for "Receiving Report Sent" field in "Examine A Shipment" module.

USER'S GUIDE

Introduction to C-LMIS

What is C-LMIS?

[Describe what the computerized LMIS is designed to do and who it is designed for.]

C-LMIS functions

[Briefly describe each software module. The functions will be described in much greater detail in section 3, so this part can be brief.]

Minimum system requirements

[List the minimum requirements for installing the computerized LMIS on the user's computer—

- CPU speed (in MHz)
- RAM
- Hard disk space
- Operating system(s)
- Any other required components (other applications or peripheral devices)].

Before You Begin [Optional]

[If there are one or more steps that must be completed prior to using the software, list them here. For example, a computerized LMIS developed by JSI listed more than fifteen steps in this section.]

Getting Started

Installation instructions

[Give instructions for installing the computerized LMIS on a computer.]

Navigation techniques and conventions

[Describe the common navigation techniques. Show common buttons or other controls, and state what they do.]

System Configuration

[List any software operations required for configuration before starting routine operations.]

C-LMIS Functions [Detailed Instructions]

[This section should contain detailed instructions for operating each module of the computerized LMIS. Up-to-date use cases can serve as headings within this section.]

Reports

[List all the reports generated by the computerized LMIS, and briefly describe what each report contains.]

TECHNICAL MANUAL

System Overview

[This is the executive summary—describe in a few paragraphs what the business context is and what the software does to support people working in that context.]

Programming Conventions

Naming conventions

[Describe standards for naming all software elements.]

Form and report conventions

[Describe standard elements for forms and reports.]

Coding guidelines

[Outline standards for writing code—declaration of variables and global variables, comments, white space, and indentation.]

Development Conventions

[Describe the standard process for developing, testing, and releasing software modifications to production. If you are not going to further develop or maintain the application, you may leave this section out.]

Database Tables

[Briefly describe each of the tables in the database and list the columns in each table.]

Program Modules

[Briefly describe each module, library, and function contained in the software. Along with the database tables section above, this should be the longest section in the manual.]

System Administration

Server configuration

[If the software operates in a client-server configuration, include instructions for setting up and maintaining the server here.]

Adding a user

[If the procedure for adding a new user involves establishing connection to a networked database or other complex procedure, describe the procedures for adding a new user. Otherwise, the steps for adding a user can remain in the user's guide.]

Administrative tools

[List and describe additional utilities or tools that may be used to modify or enhance the LMIS.]

TRAINING CURRICULUM

Goal

Participants will understand the inputs and outputs of the software program (C-LMIS) and its use in monitoring logistics activities.

Learning Objectives

After completing this session, participants will be able to:

1. Explain what C-LMIS is.
2. Start a C-LMIS database for a new program.
3. Input and edit data in the C-LMIS software.
4. Print C-LMIS reports and graphs.
5. Interpret and explain data in reports and graphs.
6. Use the C-LMIS Users Guide as a reference in using the C-LMIS software.
7. Explain the usefulness of C-LMIS to program managers.

Time

240 minutes, including break, for activities 1-8

60 minutes for activities 9-10

Materials

- C-LMIS Users Guide, version 1.41 for each participant
- Computers with C-LMIS version 1.41 installed
- Computer projection equipment, if available
- Overhead projector and screen
- C-LMIS version 1.4 program diskettes for each participant
- Candy or some prizes for report game

Overheads

1. What C-LMIS Monitors
2. What C-LMIS Can Do For You
3. Sample Reports and Graphs (Handout 5)

TRAINING CURRICULUM (CONT.)

Handouts

1. Job Aid: Principal Menu
2. Job Aid: Update Menu—Primary (top)
3. Job Aid: Update Menu—Program Parameters (bottom)
4. Job Aid: Graphs Menu
5. Job Aid: Reports Menu
6. HELP! Cheaters Quiz
7. Kaamanland Program Information
8. Kaamanland Data Exercise, Part I
9. Kaamanland Data Exercise, Part II
10. Sample C-LMIS Reports and Graphs
11. Report Game Questions and Answers

Notes

1. Activities 1-8 will take place in the hands-on computer training room. Activities 9-10 will take place after returning to the regular training room.
2. Distribute User's Guide the night before the session so participants can familiarize themselves with the guide format and with the features of the software.
3. Unit costs for non-USAID products are not in place. In order to avoid the unit cost pop-up window during the exercise, put in a unit cost for DFID before beginning to work with participants in the exercise OR plan to explain this part when it occurs.

Learning Activities Summary

Title	Type	Time
Overview of the C-LMIS Software	Lecturette	10
Walk Through of Features: Graphs and Reports	Interactive demonstration	50
Using Help	Quiz	15
Entering Program Data	Hands-on with computers	45
Entering Data for a New Program	Hands-on with computers	45
Printing & Checking Reports	Discussion	15
Entering more data for a new program	Individual exercise	40
Sample C-LMIS Graphs & Reports	Homework	5
Review of the C-LMIS Reports	Game	45
Usefulness of C-LMIS Reports	Discussion	15